INTERACTIONS ON COMPLEX NETWORKS: INFERENCE ALGORITHMS AND APPLICATIONS

A Dissertation

Presented to

the Faculty of the Department of Computer Science

University of Houston

In Partial Fulfillment

of the Requirements for the Degree

Doctor of Philosophy

By

Huy Nguyen

May 2013

INTERACTIONS ON COMPLEX NETWORKS: INFERENCE ALGORITHMS AND APPLICATIONS

Huy Nguyen

APPROVED:

Rong Zheng, Chairman
Dept. of Computer Science, Univ. of Houston

Zhu Han
Dept. of Electrical and Computer Engineering,
Univ. of Houston

Jehan-Francois Paris
Dept. of Computer Science, Univ. of Houston

Christoph Eick
Dept. of Computer Science, Univ. of Houston

Weidong (Larry) Shi
Dept. of Computer Science, Univ. of Houston

Dean, College of Natural Sciences and Mathematics

Acknowledgements

First, I would like to thank my advisor, Dr. Rong Zheng for her advice and support. This dissertation would not have been possible without her and without the freedom, encouragement, and caring she has given me over the last four years. It is not often that one finds an advisor who is always available to listen to all the problems and issues that come up during the course of research. Her dedication and advice have taught me innumerable lessons and provided insights on the workings of academic research in general.

I have a wonderful group of coauthors and collaborators. Each of them deserves my gratitude: Dr. Zhu Han, Dr. Gabriel Scalosub, Guanbo Zheng, Nam Nguyen, Mohammad Esmalifalak, Yi Huang, and Arun Chhetri. I would especially like to thank Dr. Zhu Han and Dr. Gabriel Scalosub for the fruitful collaboration that results in indispensable parts of this dissertation. Working with them has been a great experience and I was able to learn a lot from their great ideas, insights, and advice.

Also, I would like thank my thesis committee members, Dr. Jehan-Francois Paris, Dr. Christoph Eick, and Dr. Larry Shi for their advice and comments. Without their help, technically and editorially, this dissertation would not be complete.

Finally, I thank my parents and my younger brother, for endless love, encouragement, advice and support. Thanks for always being there, sharing, caring, and helping me overcome any difficulties.

INTERACTIONS ON COMPLEX NETWORKS: INFERENCE ALGORITHMS AND APPLICATIONS

An Abstract of a Dissertation

Presented to

the Faculty of the Department of Computer Science

University of Houston

In Partial Fulfillment

of the Requirements for the Degree

Doctor of Philosophy

By

Huy Nguyen

May 2013

Abstract

Complex networks are ubiquitous — from social and information systems to biological and technological systems. Such networks are platforms for interaction, communication, and collaboration among distributed entities. Studying and analyzing observable network interactions are therefore crucial to understand the hidden complex network properties.

However, with pervasive adoption of the Internet and technology advancements, networks under study today are not only substantially larger than those in the past, but are often highly distributed over large geographical areas. Along with this massive scale, the volume of interaction data also presents a serious challenge to network analysis and data mining techniques. This dissertation focuses on developing inference solutions to complex networks from different domains and applying them in solving practical problems in information and social sciences.

In the first part of the dissertation, we propose Binary Independent Component Analysis with OR Mixtures (bICA), an inference algorithm specialized for communication networks that can be formulated as a bipartite graph. Then we apply bICA and its variants to solve a wide range of networking problems, ranging from optimal monitoring and primary user separation in wireless networks to multicast network tree topology inference. Evaluation results show that the methodology is not only more accurate than previous approaches, but also more robust against measurement noise.

In the second part, we extend our study to the online social networking domain, where the networks are both massive and dynamic. We conduct an extensive

analysis on Twitter and associated influence ranking services. Several interesting discoveries have been made, which challenge some of the basic assumptions that many researchers made in the past. We also investigate the problem of finding the set of most influential entities on social networks given a limited budget. Experiments conducted on both large-scale social networks and synthetically generated networks demonstrate the effectiveness of the proposed solution.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview

Many systems across numerous and diverse domains are naturally represented as networks. A network is a system of interconnected entities typically represented mathematically as a graph (i.e., a set of nodes and a set of links among the nodes). With the pervasive use of the Internet, and the proliferation of mobile phones and location-aware devices, there is a ever increasing availability of network data. For instance, many Web-based services, such as online social networks, produce large amounts of data on interactions and associations among individuals (e.g., Facebook claims to have 1 billion users by October 2012 [18], occupying nearly half of the world's roughly 2.5 billion Internet users). Mobile phones and location-aware devices produce copious amounts of data on both communication patterns and physical proximity among people (e.g., [41, 79]). In the domain of biology, from neurons to

proteins to food webs, it is now possible to access to large networks of associations among various entities necessitating novel methodologies to analyze and understand these network data. The term "complex network" was therefore introduced to describe networks with non-trivial topological features – features that do not occur in simple networks such as lattices or random graphs but often emerge in real graphs. A concrete definition of *complex network* is not currently available in literature, however, it is widely believed that a complex network should both be scale-free (with node degree distribution following power law distribution) and have small diameter (small shortest path distance).

Interactions among network entities have various manifestations in different application domains. It could be a data packet relayed from one router to another, a signal from the access point to the end-user in communication networks, or a piece of information spread among friends in a social network. Studying the network interaction leads to a better understanding of the underlying network properties and dynamic behaviors on it. Unfortunately, while some interactions are detectable and can be fully captured, like data transmissions between computers in a local network; most of them are hidden, or just partially observable. For example, how a probe packet is delivered from the root to end receivers in a large multicast network, or how an interesting story spreads from the source to many readers on the blog sphere is typically very difficult, if not impossible, to be fully observed. Another challenge is often times, the amount of observable data surpasses processing power in real time. The problem is popular in network monitoring where network administrators need to apply random sampling on the trace data. As another example, effectively detecting

and removing spam tweets on Twitter is a daunting task for data analysts.

The main objective of our research is to study and analyze network interactions to gain a better understanding of its hidden structures and dynamic behaviors. Although there exist a substantial body of work on interactions among networked entities in statistical literature and in social sciences, our objective is different. We focus on devising robust methodologies to make inference from the observable interaction data and apply them to solve real-world problems.

1.2 Main contributions

Our contribution in this dissertation is two-fold. First we study the inference problem on communication networks. We observe in many existing problems, the underlying network could be characterized as a bipartite graph, where the network itself can be partitioned into two sets of interacting nodes. We propose Binary Independent Component Analysis with OR Mixtures (bICA), an inference algorithm that reveals the interaction between hidden and observable nodes on a bipartite graph. Then we study the interactions on social networks, which are not only massive in scale, but also structurally dynamic. We collect a large dataset from Twitter and conduct a quantitative study, which reveals many interesting properties of the underlying network. Furthermore, we define the Budgeted Influence Maximization problem on social networks and propose a greedy solution to identify the set of most influential nodes.

1.2.1 Inference on communication networks

We study inference problems on a diverge range of communication networks and realize that many of them can be characterized as a special type of graphs: bipartite graphs, where nodes can be divided into two disjoint sets such that every edge connects a node in the first set to one in the second set. If each node in the network is associated with a variable, the network consists of one "hidden set" containing latent variables and one "observable set" of observable variables. The problem of interest is, given the time series of observable data, how to infer the connection between the two sets of nodes and parameters of the hidden variables. If the relationship between the two sets is a linear relationship, then the problem can be solved using standard independent component analysis (ICA) techniques. However, the inference becomes challenging if the variables are boolean and the edges are associated with disjunctive operators. We propose bICA which is an inference method for binary variables on bipartite graphs and apply it to solve problems from various application domains, including:

- Optimal monitoring in multichannel wireless networks
- Primary user separation in cognitive radio networks
- Multicast tree topology inference in wired networks
- Blind identification of wireless transmission technologies in wide spectrum

1.2.2 Inference on social networks

Monitoring and analyzing social networks is inherently more challenging than traditional ones because of the massive scale of these networks and innate features such as scale-free and small-diameter. Our study focuses on the problem of information diffusing in social networks, where a piece of information originated from a set of nodes spreads itself over the edges of the underlying network.

To gain a comprehensive first-hand understanding on how information is spread, we first crawl the networks of different communities on Twitter, along with interactions among the nodes. Our dataset contains more than 20.5 million user profiles, 420.2 million social relationships and 105 million interactions (i.e., retweets, replies) between these users. We also obtain the digital influence score of more than 18.3 million users (89.4%) from Klout and PeerIndex. Analysis on the resulting dataset shows that some of the widely accepted assumptions in the research community do not hold on Twitter. In response, we propose a new information diffusion model on Twitter.

We next investigate the influence maximization (IM) problem, where the goal is to pick a set of k most influential entities on the social sphere. The IM problem is particularly relevant in viral marketing where a vendor needs to target a limited set of consumers who can spread words regarding a product to maximize that product's adoption. Another application of the IM problem is in the field of epidemiology where we need quickly identify and immunize the set of key nodes that if affected can potentially corrupt the network. As a generalization of the IM problem, we introduce

the Budgeted Influence Maximization (BIM) problem, where "cost" is associated with each node. A greedy algorithm that can effectively solve the BIM problem on large-scale networks is proposed and evaluated on real-world and synthetic datasets.

1.3 Organization of the dissertation

This dissertation is structured as follows: In Chapter 2, we provide definitions of the basic concepts and a brief survey on related literature. Chapter 3 focuses on our contribution on communication networks. We describe the inference methodology on bipartite graphs (bICA) and its applications in various networking problems. Chapter 4 discusses the study on social networks. We first describe the process to collect Twitter social network dataset and present the key findings from analyzing the dataset. Then we rigorously define the budgeted influence maximization problem and prove its hardness. A greedy algorithm is proposed and evaluated on both real and synthetic datasets. Finally, we summarize our contributions and discuss potential directions for future research.

Chapter 2

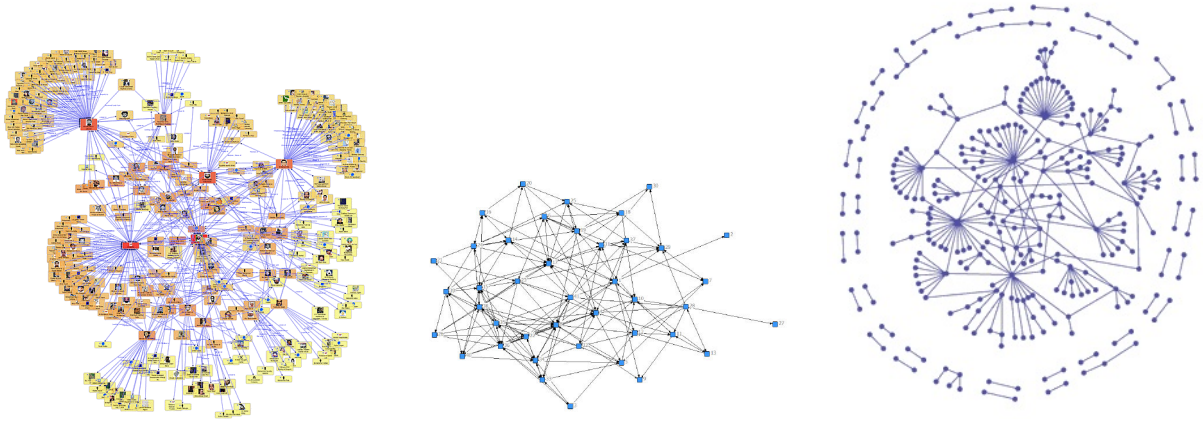
Background

In this chapter, we first present the problem formulation along with the basic definitions and notations used throughout the dissertation. Several important properties of complex networks are also discussed. Finally, we give a brief review of the related literature on network inference.

2.1 Problem formulation

2.1.1 Definitions and notations

$G = (V, E)$ is a network or graph where V is the set of *vertices* or *nodes* and $E \subseteq V \times V$ is a set of *links* or *edges*. We denote by N the number of nodes ($N = |V|$) and by M the number of edges ($M = |E|$). We use the terms *network* and *graph* interchangeably. Note that a network here is *static* which is different from a *dynamic* network – a



(a) Social network (b) Collaboration network (c) Protein network

Figure 2.1: Three networks from different domains: (a) Nodes are people, and edges represent friendship relationships. (b) Nodes are people, and edges represent work collaborations. (c) Nodes are proteins and edges represent interactions.

network with varying structure and parameters at each time step [23].

The nodes and links of a graph may be used to represent a variety of systems. In a social network, for instance, nodes may represent people and links may represent relationships (e.g., friendships), interactions (e.g., emails transmitted, physical proximity), or even similarity (e.g., similar books purchased). Likewise, in biological systems, nodes may represent neurons or, say, proteins and the links may represent neuronal connections or protein interactions (respectively). Figure 2.1 shows networks from three different domains.

A graph may be *undirected* or *directed*. The edges of an undirected graph have no explicit direction; they can be used to represent bidirectional relations between the connected nodes. A directed graph has unidirectional or directed edges, implying interaction directed from one node to the other, but not vice versa. Directed edges

provide the mechanism for representing causality. Graphs may also be either *cyclic* or *acyclic*. An acyclic graph contains no cycle, that is, there doesn't exist a path starting from any vertex looping back to the that same node. On the other hand, a cyclic graph must contain at least one cycle. In this dissertation, we focus the discussion on *general directed graphs* as depicted in the Figure 2.2 since they are the best representation of realistic networks. Together with probabilities associated on the edges, they are similar to Bayesian networks, except that there may be cycles in the graph. Probabilistic techniques on directed acyclic graphs, however, serve as the basis in the proposed methodologies on the general directed graph.

Each node in the graph is associated with a variable that can be in a finite number of states. We denote by x_i the variable representing the state of node i. Associated with the graph is a set of conditional probabilities: for example, consider the network as depicted in Figure 2.2, we denote by $p(x_4|x_2)$ the conditional probability of x_4 given x_2. In this case, we say that the node 2 is the "parent" of the node 4 because x_4 is conditionally dependent on x_2. Some nodes like node 8 might have more than one parent, in which case we define their conditional probabilities in term of all their parents; thus we write $p(x_8|x_5, x_6)$ for the conditional probability of x_8. For nodes such as node 1 or node 2, which do not have any parent, we introduce probabilities $p(x_1)$ and $p(x_2)$ that are not conditioned on any other variables.

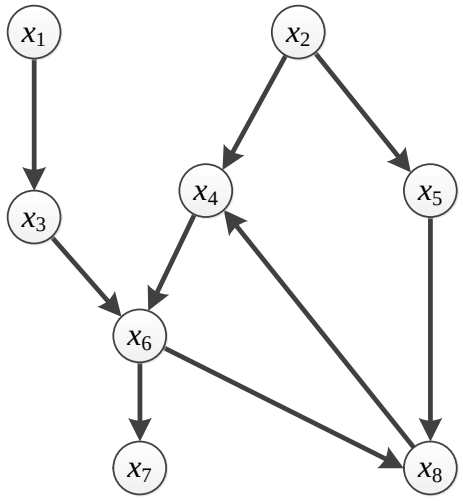


Figure 2.2: A general directed graph with 8 nodes. Note that the graph has a cycle containing $\{x_4, x_6, x_8\}$

2.1.2 Network interaction

Define the *network interaction log* as a set of triples (u, v, t) where at time t, node u performs an action towards node v, and u is the source and v is the destination. We consider one-on-one relationship. A one-to-many interaction, such as broadcasting a message or spreading an idea, can be thought of as a collection of one-on-one interactions. We also assume that the interaction (u, v, t) is possible only when there is a direct connection from u to v (i.e., the edge $(u, v) \in E$). Some examples of interactions on different networks are listed as follows:

On a wired network: Packet forwarding by intermediate network entities (router, switch, hub, etc.)

On a wireless network: Signal transmissions between network entities (access point, end-user, sniffer, etc.)

On a social network: Information (tweet, status, message, blog, idea, etc.) passed from one user to another.

Collecting interaction log is feasible for small scale networks. However, it is challenge to obtain complete information on larger networks. Furthermore, the interactions are sometimes hidden, or just partially observable. For example, it is possible to directly observe when nodes become infected with a virus; on the other hand, observing individual interaction (from whom it was infected) is very difficult.

2.2 Characteristics of complex networks

Research over the past few years has identified properties universal in many real-world networks from various domains. In this section, we discuss some fundamental patterns that have been discovered in real complex networks, including heavy-tailed degree distribution, small diameter, and community structure.

2.2.1 Heavy-tailed degree distribution

The degree-distribution of a graph follow a power law if the number of nodes N_d of degree d is given by $N_d \propto d^{-\alpha}$, where $\alpha > 1$ is called the power law degree exponent. For most real-world datasets, we have $2 < \alpha < 3$. Most of large real-world networks exhibit heavy-tailed or power law degree distributions, and are thus often called

scale-free networks. This property is important as it differentiate real networks form "randomly generated" networks.

2.2.2 Small diameter

The concept of a small-world network was first mentioned by Milgram *et al.* in [87] where the authors presented and empirically validated the hypothesis that any two persons on the planet are separated by at most six degrees of separation. In general, small-diameter or small-world networks are graphs in which most nodes are not neighbors of one another, but most nodes can be reached from every other through small number of hops or steps. Formally, let L be the distance (of the shortest path) between two random nodes and N be the number of nodes in the network, we have $L \propto \log(N)$ [15] or $L \propto \log\log(N)$ [36, 35].

2.2.3 Community structure

A network is said to have community structure if the nodes can be easily grouped into sets of nodes such that more edges are present among members of a set than between most of its members and other nodes on the network [52]. The problem of community identification is often formulated as an unsupervised learning problem, a form of clustering or graph partitioning with the objectives to partition the network into *disjoint* but sometime also *overlapping* sets of nodes. Finding overlapping communities is more difficult and requires more sophisticated techniques [67]. It is also observed that communities have a recursive structure, where bigger communities can

further be split into smaller and smaller communities [33, 105]. Interpretation of "a community" depends on the application domain. It could be organization units in social networks, functional modules in biological networks, scientific disciplines in collaboration networks between scientists, etc.

2.3 Network inference

As previously mentioned, complete structural information or interaction traces is usually not available for complex network analysis. A technique that can be used to alleviate these difficulties is *network inference*, where the problem is to estimate the model or the interactions of different elements in the network, given the observed data. There are two types of network inference: structure inference and parameter inference. We briefly survey in this section related literature of each category.

2.3.1 Structure inference

Inducing the network structure is known to be a challenging problem due to enormity of the search space. The number of possible network structures grows super-exponentially with the number of nodes.

2.3.1.1 Communication networks

Structure inference on communication networks is referred to as the *network topology identification* problem. When the network topology is unknown, tools such as

traceroute [10] or ping can be used in an attempt to identify it. These tools assume the network is instrumented properly and the network elements are cooperating in revealing themselves. These conditions are often not met in the modern Internet area due to many reasons, like the lack of motivation to cooperate (since it consumes computing and storage resources), or security issues (risk of exposing network details), etc. It is therefore desirable to develop methods to estimate the network topology using only measurements taken at network end-points.

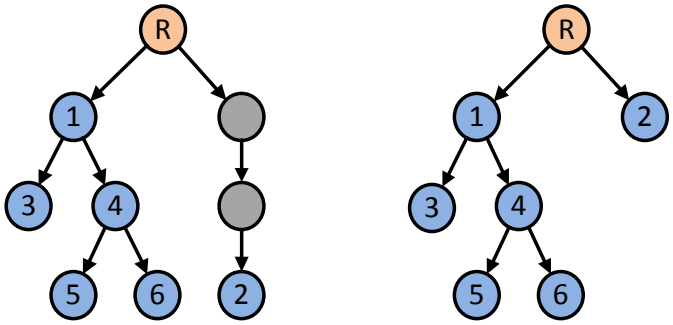


Figure 2.3: Physical topology (left) and corresponding logical topology (right). R is the root node (source) and 2,3,5,6 are the leaf nodes (receivers). The dark unnumbered nodes are devices where no branching of traffic occurs and therefore can not be detected.

The problem is illustrated in Figure 2.3. We consider a single source that is communicating with multiple receivers (denoted by the root node R). The physical topology can be represented by a directed graph, where each node represents a network device and each edge represents a connection between the devices. Since we limit ourselves to the availability of end-to-end measurements only, it is therefore

only possible to identify the "logical topology" of the network. In the logical topology, each vertex represents a physical network device where the traffic branching occurs.

There are two major categories of approaches in network topology identification: using *unicast* and *multicast* probe packets. Ratnasamy *et al.* [100] were the first to demonstrate that correlations in multicast loss measurements could be used to reconstruct the logical topology. Later, Duffield *et al.* [43] rigourously established the correctness of their proposed algorithm and developed a more general framework in which other measurements, such as delay variance, could be used. Most recently, we revisited the problem of multicast tree topology inference in [94] and proposed a new solution based on independency analysis between receivers. The proposed method was shown to converge much faster and obtain higher accuracy than the state-of-the-art solutions.

Due to the dominance of *unicast* networks, there is a larger amount of literature focuses on using *unicast* packets to infer the network topology. Duffield *et al.* conducted a series of study on the topic. In [45, 46, 42], the authors adapted the multicast inference techniques proposed previously to perform inference of internal network characteristics from unicast end-to-end measurements. The idea is to construct composite probes of unicast packets whose collective statistical properties closely resemble those of a multicast packet. Nguyen *et al.* also conducted multiple studies to investigate the problem. In [95], the authors proposed a new method by exploiting the second-order moments of end-to-end flows. In [92], a solution to solving systems of equations that do not have a unique solution was proposed. And

finally in [51], the authors introduced a novel network tomographic tool based on their previous findings.

Beside the literature on the traditional topology inference problem, many other studies using *unicast* packets explored different aspects of the problem, like multiple-source, multiple-destination systems [99, 98], optimal probing scenario for unicast packets [60], packet loss time correlation modeling [21], etc.

2.3.1.2 Social networks

On social networks, the problem becomes inferring the network of diffusion and influence. The problem is motivated from the blogging communities, where bloggers usually copy interesting articles without properly citing the source. In this case, capturing when a node publishes the information is easy, but it is not possible to observe the transmission (from whom the information came). The above problem is illustrated in Figure 2.4.

Leskovec *et al.* conducted several studies to model the propagation of information on the blogosphere. In [55], they converted the edge inference problem to be a combinatorial optimization problem and proposed some useful heuristics to handle large scale data. Later in [116], the authors suggested techniques to improve the inference result when only a fraction of the complete data is available. In [103], they tried to infer the node influence from the action log, given that the network graph is unobserved.

A variance of the structure inference problem above is the hierarchy inference

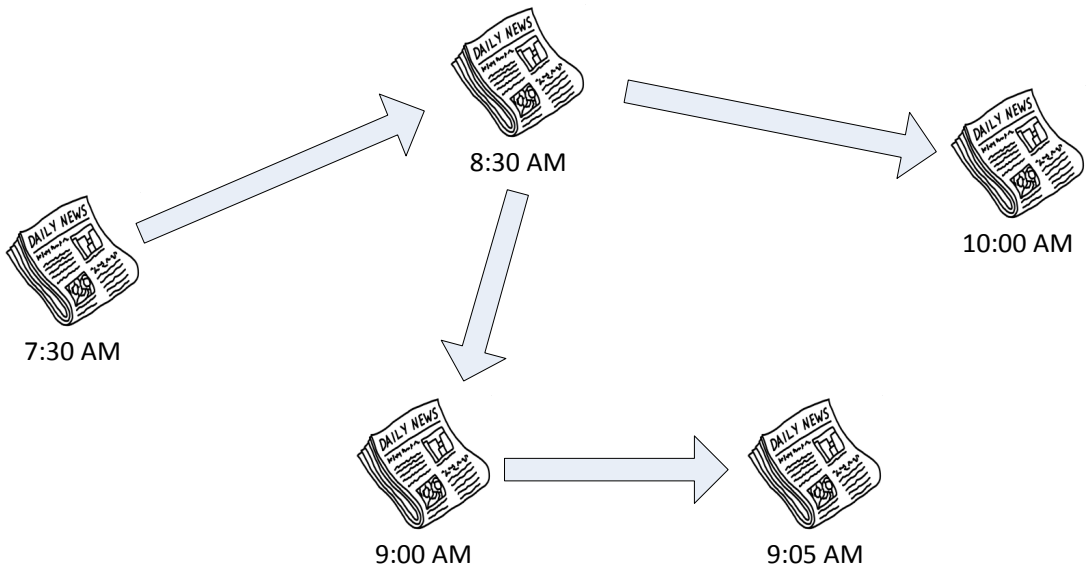


Figure 2.4: Illustration of information spreading on blogosphere. Actual transmissions (edges) are usually not observable.

problem, where the goal is to find the hidden stratification structure of the network. Recent studies showed that hierarchy does exists on many online social networks. Rowe *et al.* [102] mined the email exchange dataset from corporations and extracted their structural organization. Maiya *et al.* [82] proposed a heuristic to infer the maximum likelihood hierarchy in social networks. Evaluations on the U.S. government dataset have showed that their algorithm works well in practice. Most recently, Gupte *et al.* [61] hypothesized that people form connections in a social network based on their perceived social hierarchy. For instance, A follows B means that B's social rank is likely higher than A. The authors defined a measure of hierarchy on *directed* online social networks and presented an algorithm to compute this measure. The

proposed solution has been verified on a variety of online social networks, including Twitter, Delicious, YouTube, Flick, etc.

2.3.2 Parameter inference

Broadly speaking, parameter inference on complex networks involves estimating network parameters based on the interaction trace collected from (usually a subset of) network nodes.

2.3.2.1 Communication networks

Two forms of parameter inference problem on communication networks have been addressed in literature: link-level parameter estimation based on end-to-end traffic measurements and receiver-sender path-level traffic intensity estimation based on link-level traffic measurements.

In *link-level parameter estimation*, the collected information usually consists of counts of packets transmitted and/or received between source and destination nodes or time delays between transmissions and receptions. The goal is to estimate the loss rate or the queuing delay on each link. A packet is dropped if it does not successfully reach the input buffer of the destination node, usually due to congestion or queuing policies. In [43], Duffield *et al.* proposed three approaches to infer multicast-tree topology and link performance using loss measurements at network end-points, including (i) a grouping estimator that exploits the monotonicity of loss rate with

increasing path length; (ii) a maximum likelihood estimator; and (iii) a Bayesian estimator. It was shown that the algorithm in (i) offers the best performance in term of accuracy and computational simplicity. Tian *et al.* proposed an inference algorithm that utilizes hop count information in topology inference [109]. Most recently, Mao *et al.* [83] proposed NetworkMD, an unsupervised learning algorithm. It proceeds in two steps: (1) to find the number of monitor clusters using SVD, and (2) to infer the network topology by non-negative matrix factorization (NMF). The proposed method can be extended to multi-source, multi-destination setups like in [99].

In path-level traffic intensity estimation, the measurement consists of counts of packets that pass through nodes in the network. In privately owned networks, the collection of such measurements is straight-forward. Based on these measurements, the goal is to estimate how much traffic originated from a node and was destined for a receiver. Such information forms the *origin-destination traffic matrix*. Bin *et al.* [119] proposed a statistical inverse algorithm for any fixed routing scheme. Later, the authors proposed another solution [22] which relies on divide-and-conquer strategy to lower the computation cost without losing accuracy. The proposed algorithm is tested on a real network with 18 nodes, showing its accuracy. More recently, Liang *et al.* [81] used a pseudo likelihood approach to solve the problem. The authors first established some statistical properties and defined the pseudo likelihood function. Then a expectation-maximization (EM) algorithm was developed to maximize the pseudo log-likelihood function. The approach was evaluated on both synthetic and real network data.

2.3.2.2 Social networks

The problem of inferring edge weight (influence) on social networks is an active research topic. To perform parameter inference, a model on how the information is spread on a social network has to be established. Two of the most widely adopted diffusion models are the linear threshold (LT) and the independent cascade (IC) models.

Linear threshold model [57]: each node in the network has a threshold $t \in [0, 1]$, drawn from some probability distribution. We also assign connection weights $w_{u,v}$ on the edges from node u to node v of the network. A node is active if the sum of connection weights from its active neighbors is greater than the threshold t.

Independent cascade model [54]: whenever a neighbor u of v is active, it will have a single chance to activate v with probability $p_{u,v}$ associated with the network edge from u to v.

Leskovec *et al.* in [55] adopted the IC model and proposed a heuristic to calculate the edge weights by considering the most likely spanning tree. Saito *et al.* [104] formulated this as a likelihood maximization problem and then apply an EM algorithm to solve it. The algorithm was shown to be highly accurate, but with poor scalability. Goyal *et al.* [56] also studied the problem of learning influence probabilities on the IC model. They focused on the time varying nature of influence, and on factors such as the influenceability of a specific user and influence-proneness of a certain action. Although there is no study devoted to the problem of studying the nodal threshold on the LT model, it can be inferred by a transformation from the edge propagation

probability on the IC model since the two models are logically equivalent [70].

Chapter 3

Interactions on Communication Networks

In this chapter, we investigate the inference problem on communication networks. Interestingly, in many problems, the network can be characterized by a bipartite graph and interactions only occur between the two disjoint sets of nodes. We propose Binary Independent Component Analysis with OR mixtures (bICA), an inference algorithm on bipartite graphs. We prove that bICA is uniquely identifiable under the disjunctive generative model, and propose a deterministic iterative algorithm to determine the connections between the latent and observable variables, as well as the distribution of the latent random variables. The inverse problem to infer the values of latent variables is also considered for noisy measurements. Finally, we apply bICA to solve practical problems from several application domains.

3.1 Introduction of bICA

Independent component analysis (ICA) is a computational method for separating a multivariate signal into additive subcomponents under the mutual statistical independence of the non-Gaussian source signals. The classical ICA framework usually assumes linear combinations of independent sources over the field of real-valued numbers $\mathcal{R}$. Consider the following generative data model where the observations are disjunctive mixtures of binary independent sources. Let $\mathbf{x} = [x_1, x_2, \ldots, x_m]^T$ be an m-dimension binary random vector with joint distribution $\mathcal{P}(\mathbf{x})$, which are observable. $\mathbf{x}$ is generated from a set of n independent binary random variables $\mathbf{y} = [y_1, y_2, \ldots, y_n]^T$ as follows,

$$x_i = \bigvee_{j=1}^{n} (g_{ij} \wedge y_j), \ i = 1, \ldots, m, \tag{3.1}$$

where $\wedge$ is Boolean *AND*, $\vee$ is Boolean *OR*, and g_{ij} is the entry in the i'th row and j'th column of an unknown binary mixing matrix $\boldsymbol{G}$. Throughout this chapter, we denote by $\boldsymbol{G}_{i,:}$ and $\boldsymbol{G}_{:,j}$ the i'th row and j'th column of matrix $\boldsymbol{G}$ respectively. For the ease of presentation, we introduce a short-hand notation for the above disjunctive model as,

$$\mathbf{x} = \boldsymbol{G} \otimes \mathbf{y}.$$

The relationship between observable variables in $\mathbf{x}$ and latent binary variables in $\mathbf{y}$ can also be represented by an undirected bi-partite graph $G = (U, V, E)$, where $U = \{x_1, x_2, \ldots, x_m\}$ and $V = \{y_1, y_2, \ldots, y_n\}$ (Figure 3.1). An edge $e = (x_i, y_j)$ exists if $g_{ij} = 1$. We will refer to $\boldsymbol{G}$ as the binary adjacency matrix of graph G.

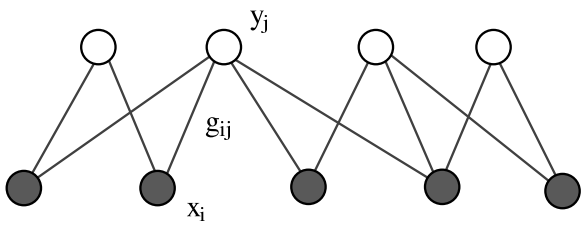


Figure 3.1: Illustration of the OR mixture model

Consider an $m \times T$ matrix $\boldsymbol{X}$ and an $n \times T$ matrix $\boldsymbol{Y}$, which are the collection of T realizations of random vector $\mathbf{x}$ and $\mathbf{y}$ respectively. The goal of bICA is to estimate the distribution of the latency random variables $\mathbf{y}$ and the binary mixing matrix $\boldsymbol{G}$ from $\boldsymbol{X}$ such that $\boldsymbol{X}$ can be decomposed into OR mixtures of columns of $\boldsymbol{Y}$.

3.2 Related work

Most ICA methods assume linear mixing of continuous signals [65]. A special variant of ICA, called binary ICA (BICA), considers boolean mixing (e.g., OR, XOR etc.) of binary signals. Existing solutions to BICA mainly differ in their assumptions of the binary operator (e.g., OR or XOR), the prior distribution of the mixing matrix, noise model, and/or hidden causes.

In [118], Yeredor considers BICA in XOR mixtures and investigates the identifiability problem. A deflation algorithm is proposed for source separation based on entropy minimization. Since XOR is addition in the Galois field of two elements (GF(2)), BICA in XOR mixtures can be viewed as the binary counterpart of classical

linear ICA problems. The number of independent random sources K is assumed to be known. Furthermore, the mixing matrix is a K-by-K invertible matrix. Under these constraints, it has been proved that the XOR model is invertible and there exist a unique transformation matrix to recover the independent components up to permutation ambiguity. Though our proof of identifiability in this chapter is inspired by the approach in [118], due to the "non-linearity" of OR operations, the notion of invertible matrices no longer applies. New proofs and algorithms are warranted to unravel the properties of binary OR mixtures.

In [69], the problem of factorization and de-noise of binary data due to independent continuous sources is considered. The sources are assumed to be continuous following beta distributions in [0, 1]. Conditional on the latent variables, the observations follow the independent Bernoulli likelihood model with mean vectors taking the form of a linear mixture of the latent variables. The mixing coefficients are assumed to be non-negative and sum to one. A variational EM solution is devised to infer the mixing coefficients. A post-process step is applied to quantize the recovered "gray-scale" sources into binary ones. While the mixing model in [69] can find many real-world applications, it is not suitable in the case of OR mixtures.

In [113], the authors introduced a noise-OR model to model dependency among observable random variables using K (known) latent factors and then developed a variational inference algorithm. The probabilistic dependency between observable vectors and latent vectors is modeled via the noise-OR conditional distribution. The dimension of the latent vector is assumed to be known and less than that of the observable.

In [37], Wood *et al.* consider the problem of inferring infinite number of hidden causes following the same Bernoulli distribution. Observations are generated from a noise-OR distribution. Prior of the infinite mixing matrix is modeled as the Indian buffet process [58]. Reversible jump Markov chain Monte Carlo and Gibbs sampler techniques are applied to determine the mixing matrix based on observations. In our model, the hidden causes are finite in size, and may follow different distributions. Streith *et al.* [107] study the problem of multi-assignment clustering for boolean data, where the observations are either drawn from a signal following OR mixtures or from a noise component. The key assumption made in the work is that the elements of matrix $\boldsymbol{X}$ are conditionally independent given the model parameters (as opposed to the latent variables). This greatly reduces the computational complexity and makes the scheme amenable to a gradient descent-based optimization solution. However, this assumption is in general invalid.

There exists a large body of work on blind deconvolution with binary sources in the context of wireless communication [38, 80]. In time-invariant linear channels, the output signal $x(k)$ is a convolution of the channel realizations $a(k)$ and the input signal $s(k)$, $k = 1, 2, \ldots, K$ as follows:

$$x(k) = \sum_{l=0}^{L} a(l)s(k-l), k = 1, \ldots, K. \tag{3.2}$$

The objective is to recover the input signal s. Both stochastic and deterministic approaches have been devised for blind deconvolution. As evident from the above expression, the output signals are linear mixtures of the input sources in time, and additionally the mixture model follows a specific structure.

Table 3.1: Comparison of related work

Algorithm	Sources	Generative model	Under/Over determined	Dimension of latent variables
[118]	Binary	Binary XOR	–	Known
[69]	Continuous	Linear	Over	Known
[113]	Binary	Noise-OR	Over	Known
[37]	Binary	Noise-OR	Under	Infinite
[107]	Binary	Binary OR	Over	Known
[50, 16]	Binary	Binary OR	Over	Unknown, try to minimize
[38, 80]	Binary	Linear	–	Known
bICA	**Binary**	**Binary OR**	**Under**	**Unknown but finite**

Literature on boolean/binary factor analysis (BFA) is also related to our work. The goal of BFA is to decompose a binary matrix $\boldsymbol{X}_{m\times T}$ into $\boldsymbol{A}_{m\times n} \otimes \boldsymbol{B}_{n\times T}$ with $\otimes$ being the *OR* mixture relationship as defined in (3.16). $\boldsymbol{X}$ in BFA is often called an attribute-object matrix providing m-dimension attributes of T objects. $\boldsymbol{A}$ and $\boldsymbol{B}$ are the attribute-factor and factor-object matrices. All the elements in $\boldsymbol{X}$, $\boldsymbol{A}$, and $\boldsymbol{B}$ are either 0 or 1. n is the number of underlying factors and is assumed to be considerably smaller than the number of objects T. BFA methods aim to find a feasible decomposition minimizing n. Frolov *et al.* study the problem of factoring a binary matrix using Hopfield neural networks [64, 50, 63]. This approach is based on a heuristic and does not provide much theoretical insight regarding the properties of the resulting decomposition. More recently, Belohlavek *et al.* propose a matrix decomposition method utilizing formal concept analysis [16]. The paper claims that optimal decomposition with the minimum number of factors are those where factors are formal concepts. It is important to note that even though BFA assumes the same disjunctive mixture model as in our work, the objective is different. While BFA tries to find a matrix factorization so that the number of factors is minimized, bICA tries

to identify *independent* components. One can easily come up an example, where the number of independent components (factors) is larger than the number of attributes. Since BFA always finds factors no larger than the number of attributes, the resulting factors are clearly dependent in this case.

Finally, [37] considers the under-represented case of fewer number of observable than latent sources with continuous noise, while [69, 107, 50, 16] deal with the over-determined case, where the number of observable variables is much larger. In this work, we consider primarily the under-represented cases that we typically encounter in data networks where the number of sensors are much smaller than the number of signal sources (i.e. users).

We summarize the aforementioned related work in Table 3.1.

3.3 Properties of bICA

In this section, we investigate the fundamental properties of bICA. In particular, we are interested in the following questions:

- **Expressiveness:** can any set of binary random variables be decomposed into binary independent components using OR mixtures?
- **Independence of OR mixtures:** for mixtures of independent sources, what is the condition that they are independent?

- **Identifiability:** given a set of binary random variables following the bICA data model, is the decomposition unique?

Expressiveness: Expressiveness of OR mixtures is limited. This can be shown through an example. Let y_1 and y_2 be two independent binary random variables with $P(y_1 = 1) = p \neq 0.5$ and $P(y_2 = 1) = q \neq 0.5$. Let $x_1 = y_1$ and $x_2 = y_1 + y_2$, where '+' is addition in the finite field GF(2). It is easy to see that x_1 and x_2 are correlated since $P(x_2 = 1) = P(y_1 = 1)P(y_2 = 0) + P(y_1 = 0)P(y_2 = 1) = q(1 - p) + p(1 - q)$, $P(x_1 = 1) = p$, while $P(x_1 = 1, x_2 = 1) = P(y_2 = 0) = 1 - q$. On the other hand, x_2 can not be decomposed into an OR mixture of y_1 and y_2. This essentially shows that OR mixtures of binary random variables only span a subset of multi-variate binary distributions. There exist correlated binary random variables (x_1, x_2 in this example) that cannot be modeled as OR mixtures of independent binary components.

Independence of mixtures: Now we turn to the second question, namely, under what condition are binary random variables that follow the OR mixture model independent. In general, pairwise independent random variables are not jointly independent. Interestingly, we show that pairwise independence implies joint independence for OR mixtures.

Theorem 1 *Let* $\mathbf{y} = [y_1, y_2, \ldots, y_n]^T$ *denote* n *statistically independent sources in GF(2), the* i*-th source having 1-probability* p_i *. Let* $\mathbf{x} = \boldsymbol{D} \otimes \mathbf{y}$*, where* $\boldsymbol{D}$ *is an* $m \times n$ *matrix (with elements in GF(2)). Let* $\eta(\mathbf{x})$ *and* $\boldsymbol{C}(\mathbf{x})$ *denote the mean and covariance (resp.) of* $\mathbf{x}$*. If:*

1. *All elements of* $\eta(\mathbf{x})$ *are nonzero and not 1's (called non-degenerate),*

2. $\boldsymbol{C}(\mathbf{x})$ *is diagonal,*

Then i) $m = n$*, and ii)* $\boldsymbol{D}$ *is a permutation matrix.*

We first establish the following lemmas.

Lemma 1 *Let* u *and* v *be two RVs in GF(2) with 1-probabilities* p_u *and* p_v *(resp.), and* $w \stackrel{\Delta}{=} u \vee v$*. If* u *and* v *are independent, non-degenerate (*$0 < p, q < 1$*) then* w *is also non-degenerate.*

Proof. Clearly, $p_w = P(w = 1) = 1 - (1 - p_u)(1 - p_v)$. Since $0 < p_u, p_v < 1$, we have $0 < p_w < 1$. ■

Lemma 2 *Consider non-degenerate independent binary random variables* y_1, y_2, y_3*. Then,* $x_1 = y_1 \vee y_2$ *and* $x_2 = y_3$ *are independent.*

Proof. To prove independence of two binary random variables x_1 and x_2, it is sufficient to show $P(x_1 = 1, x_2 = 1) = P(x_1 = 1)P(x_2 = 1)$.

$$\begin{aligned} P(x_1 = 1, x_2 = 1) &= P(y_1 = 1, y_2 = 1, y_3 = 1) \\ &+ P(y_1 = 1, y_2 = 0, y_3 = 1) \\ &+ P(y_1 = 0, y_2 = 1, y_3 = 1) \\ &= P(y_1 = 1)P(y_2 = 1)P(y_3 = 1) \\ &+ P(y_1 = 1)P(y_2 = 0)P(y_3 = 1) \\ &+ P(y_1 = 0)P(y_2 = 1)P(y_3 = 1) \\ &= P(x_1 = 1)P(x_2 = 1) \end{aligned} \tag{3.3}$$

■

Similarly, we can show the following result.

Lemma 3 *Consider non-degenerate independent binary random variables y_1, y_2, y_3. Then, $x_1 = y_1 \vee y_2$ and $x_2 = y_1 \vee y_3$ are correlated.*

Now we are in the position to prove Theorem 1.

Proof. (Proof of Theorem 1) We prove by contradiction. The essence of the proof is similar to that in [118]. Let us assume now that $\boldsymbol{D}$ is a general matrix, and consider any pair x_k and x_l $(k \neq l)$ in $\mathbf{x}$. x_k and x_l are OR mixtures of respective subgroups of the sources, indexed by the 1-s in $\boldsymbol{D}_{k,:}$, and $\boldsymbol{D}_{l,:}$, the k-th and l-th rows (respectively) of $\boldsymbol{D}$. These two subgroups consist of, in turn, three other subgroups (some of which may be empty):

1. Sub-group 1: Sources common to $\boldsymbol{D}_{k,:}$ and $\boldsymbol{D}_{l,:}$. Denote the OR mixing of these sources as u;

2. Sub-group 2: Sources included in $\boldsymbol{D}_{k,:}$ but excluded from $\boldsymbol{D}_{l,:}$. Denote the OR mixing of these sources as v_1;

3. Sub-group 3: Sources included in $\boldsymbol{D}_{l,:}$ but excluded from $\boldsymbol{D}_{k,:}$. Denote the OR mixing of these sources as v_2.

In other words, $x_k = u \vee v_1$ and $x_l = u \vee v_2$. By applying Lemma 2 iteratively, we can show that v_1 and v_2 are independent and non-degenerate. Furthermore, if $u \neq 0$, then u is independent of v_1 and v_2. From Lemma 3, we show that x_k and x_l are

correlated. This contradicts with the condition that $\boldsymbol{C}(\mathbf{x})$ is diagonal. This implies that $u = 0$. Therefore, the two rows $\boldsymbol{D}_{k,:}$ and $\boldsymbol{D}_{l,:}$ do not share common sources, or, in other words, there is no column j in $\boldsymbol{D}$ such that both $\boldsymbol{D}_{k,j}$ and $\boldsymbol{D}_{l,j}$ are both 1. There are only m such columns. Thus, $m = n$. Furthermore, $\boldsymbol{D}$ is a permutation matrix. ■

Theorem 1 necessarily implies the following result:

Corollary 1 *Let* $\mathbf{x} = \boldsymbol{G} \otimes \mathbf{y}$ *for some* $\boldsymbol{G}$ *and independent non-degenerate sources* $\mathbf{y}$. *Then, if elements of* $\mathbf{x}$ *is non-degenerate and pair-wise independent, the elements in* $\mathbf{x}$ *are jointly independent.*

Identifiability: Let $\mathbf{x} = [x_1, \ldots, x_m]^T$. Define the set

$$Y(\mathbf{x}) = \{\mathbf{y} \mid \bigvee_{j=1}^{n} (g_{ij} \wedge y_j) = x_i, \forall i = 1, \ldots, m\}.$$

Therefore,

$$\begin{aligned} \mathcal{P}(\mathbf{x}) &= \mathcal{P}(\mathbf{y} \in Y(\mathbf{x})) = \textstyle\sum_{\mathbf{y} \in Y(\mathbf{x})} \mathcal{P}(\mathbf{y}) \\ &= \textstyle\sum_{\mathbf{y} \in Y(\mathbf{x})} \prod_{i=1}^{n} p_i^{y_i} (1 - p_i)^{1 - y_i} \end{aligned} \tag{3.4}$$

where $\mathcal{P}(\mathbf{y})$ is the joint probability of $\mathbf{y}$, and $p_i \triangleq \mathcal{P}(y_i = 1)$. The last equality is due to the independence among y_i's.

To see whether $\mathbf{y}$ is uniquely identifiable from $\mathbf{x}$, we first restrict $\boldsymbol{G}$ such that it has no identical columns, namely, each y_j contributes to a unique set of x_i's. Otherwise, if $\boldsymbol{G}_{:,i}$ and $\boldsymbol{G}_{:,j}$ are identical, we can merge y_i and y_j by a new component

corresponding to $y_i \vee y_j$. Under the restriction, we can initialize $n = 2^m - 1$ and $\boldsymbol{G}$ of dimension $m \times 2^m - 1$ with rows being all possible n binary values. The $\boldsymbol{G}$ matrix corresponds to a complete bipartite graph, where an edge exists between any two vertices in U and V, respectively. For a random variable $y_j \in V$, its neighbors in U is given by the non-zero entries in $\boldsymbol{G}_{:,j}$. Thus, at most $2^m - 1$ independent components can be identified. Given the distribution of random variables $\mathbf{x} \in \{0,1\}^m$, $2^m - 1$ equations can be obtained from (3.4). As there are at most $2^m - 1$ unknowns (i.e., $p_i, i = 1, \ldots, n$), the probability of y_j can be determined if a solution exists. To see that the solution uniquely exists, we present a constructive proof as follows.

Let $\boldsymbol{g}_k$, $k = 1, \ldots, 2^m - 1$ be an m-dimension binary column vector, and the degree of $\boldsymbol{g}_k$, $d(\boldsymbol{g}_k)$ is the number of ones in $\boldsymbol{g}_k$. Define the frequency function $\mathcal{F}_k = \mathcal{P}(\mathbf{x} = \boldsymbol{g}_k) = \mathcal{P}(x_i = g_{ik}, i = 1, \ldots m)$. For each $\boldsymbol{g}_k$, we associate it with an independent component y_k. The goal is to show that $p_k \stackrel{\Delta}{=} \mathcal{P}(y_k = 1)$ can be uniquely decided. Starting from $\boldsymbol{g}_k$ with the lowest degree, the derivation proceeds to determine p_k's with increasing degree in $\boldsymbol{g}_k$.

***Basis*:** It is easy to show that $\mathcal{F}_0 = \prod_{j=1}^{2^m-1}(1 - p_j)$. Since p_k's are non-degenerate, $\mathcal{F}_0 > 0$. For k, s.t., $d(\boldsymbol{g}_k) = 1$, we have

$$\mathcal{F}_k = p_k \prod_{j=1, j\neq k}^{2^m-1} (1 - p_j).$$

Therefore,

$$p_k = \frac{\mathcal{F}_k}{\mathcal{F}_k + \mathcal{F}_0}\mathcal{F}_0. \tag{3.5}$$

***Induction*:** Define $\boldsymbol{g}_i \prec \boldsymbol{g}_j$ if $\boldsymbol{g}_i \neq \boldsymbol{g}_j$, and $\forall l$, s.t., $g_{li} = 1$, $g_{lj} = 1$. Let S_k be the set of indices i's, s.t., $\mathcal{F}_i \neq 0$ and $\boldsymbol{g}_i \prec \boldsymbol{g}_k$, $\forall i \in S$. If $S_k = \emptyset$, then (3.5) applies.

Otherwise, we have

$$\begin{aligned}
& \mathcal{F}_k \\
= \; & \textstyle\prod_{j\notin S_k, j\neq k}(1-p_j) \times (p_k + \\
& (1-p_k)\textstyle\sum_{B\subset S_k, \bigvee_{i\in B}\boldsymbol{g}_i=\boldsymbol{g}_k}\prod_{i\in B} p_i \prod_{i\in B-S_k}(1-p_i)) \\
= \; & \tfrac{\mathcal{F}_0}{(1-p_k)\prod_{j\in S_k}(1-p_j)} \times (p_k + \\
& (1-p_k)\textstyle\sum_{B\subset S_k, \bigvee_{i\in B}\boldsymbol{g}_i=\boldsymbol{g}_k}\prod_{i\in B} p_i \prod_{i\in B-S_k}(1-p_i)) \\
= \; & \tfrac{\mathcal{F}_0}{\prod_{j\in S_k}(1-p_j)} \times (\tfrac{p_k}{1-p_k} + \\
& \textstyle\sum_{B\subset S_k, \bigvee_{i\in B}\boldsymbol{g}_i=\boldsymbol{g}_k}\prod_{i\in B} p_i \prod_{i\in B-S_k}(1-p_i)).
\end{aligned}$$

where $\bigvee_{i\in B}\boldsymbol{g}_i$ indicates the entry-wise OR of $\boldsymbol{g}_i$'s for $i \in B$. Let us define $L_k \triangleq \sum_{B\subset S, \bigvee_{i\in B}\boldsymbol{g}_i=\boldsymbol{g}_k}\prod_{i\in B} p_i \prod_{i\in B-S}(1-p_i))$. Then,

$$p_k = \frac{\mathcal{F}_k \prod_{i\in S_k}(1-p_i) - \mathcal{F}_0 L_k}{\mathcal{F}_0 + \mathcal{F}_k \prod_{i\in S_k}(1-p_i) - \mathcal{F}_0 L_k}. \tag{3.6}$$

It is easy to verify that when the y_i's are non-degenerate, all the denominators are positive. This proves that a solution to (3.4) exists and is unique. However, direct application of the construction suffers from several problems. First, all $\mathcal{F}_k$'s need to be computed from the data, which requires a large amount of observations. Second, the property that $\mathcal{F}_0 \neq 0$ is very critical in estimating p_k's. When $\mathcal{F}_0$ is small, it cannot be estimated reliably. Third, enumerating S_k for each k is computationally prohibitive.

3.4 The inference algorithm

We initialize $\boldsymbol{G}$ to be an $m \times 2^m$ adjacent matrix for the complete bipartite graph. Furthermore, the columns of $\boldsymbol{G}$ are ordered such that $g_{kl} = 1$ if $l \wedge 2^k = 1$, for $k = 1, \ldots, m$, where $\wedge$ is the bit-wise AND operator. As an example, when $m = 3$ we have:

$$\boldsymbol{G} = \begin{bmatrix} 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}$$

If the active probability of the l'th component $p_l = 0$, this implies the corresponding column $\boldsymbol{G}_{:,l}$ can be removed from $\boldsymbol{G}$. Before proceeding to the details of the algorithm, we first present a technical lemma.

Lemma 4 *Consider a set* $\mathbf{x} = [x_1, x_2, \ldots, x_{h-1}, x_h]^T$ *generated by the data model in* (3.16)*, i.e.,* $\exists$ *binary independent sources* $\mathbf{y}$*, s.t.,* $\mathbf{x} = \boldsymbol{G} \otimes \mathbf{y}$*. The conditional random vector* $\mathbf{x}_{x_h=0} = [x_1, x_2, \ldots, x_{h-1} | x_h = 0]^T$ *corresponds to the vector of the first* $h-1$ *elements of* $\mathbf{x}$ *when* $x_h = 0$*. Then,* $\mathbf{x}_{x_h=0} = \boldsymbol{G}' \otimes \mathbf{y}'$*, where* $\boldsymbol{G}' = \boldsymbol{G}_{:,1\ldots2^{h-1}}$ *(i.e. the first* 2^{h-1} *columns of* $\boldsymbol{G}$*) and* $\mathcal{P}(y'_l = 1) = \mathcal{P}(y_l = 1)$ *for* $l = 1, \ldots, 2^{h-1}$*.*

Proof. We first derive the conditional probability distribution of the first $h-1$ observation variables given $x_h = 0$,

$$\begin{aligned}
& \mathcal{P}(x_1, x_2, \ldots, x_{h-1} \mid x_h = 0) \\
= \; & \mathcal{P}(x_1, x_2, \ldots, x_{h-1} \mid x_h = 0)\mathcal{P}(x_h = 0) \\
\overset{(a)}{=} \; & \sum_{\mathbf{y} \in Y(\mathbf{x})} \prod_{l=1}^{2^h-1} p_l^{y_l}(1-p_l)^{1-y_l} \\
= \; & \sum_{\substack{\mathbf{y}_{1..2^{h-1}} \in Y(\mathbf{x}_{1..h-1}) \\ y_l = 0, \forall g_{hl} = 1}} \prod_{g_{hl}=0} p_l^{y_l}(1-p_l)^{1-y_l} \prod_{g_{hl}=1} (1-p_l).
\end{aligned} \tag{3.7}$$

(a) is due to (3.4). Since $\mathcal{P}(x_h = 0) = \prod_{g_{hl}=1} (1-p_l)$, we have

$$\begin{aligned}
& \mathcal{P}(x_1, x_2, \ldots, x_{h-1} \mid x_h = 0) \\
= \; & \sum_{\mathbf{y}' \in Y(\mathbf{x}_{1:h-1})} \prod_{l=1}^{2^h-1} (p'_l)^{y'_l}(1-p'_l)^{1-y'_l} \\
= \; & \sum_{\substack{\mathbf{y}_{1,\ldots,2^{h-1}} \in Y(\mathbf{x}_{1,\ldots,h-1}) \\ y_l = 0, \forall g_{hl} = 1}} \prod_{g_{hl}=0} p_l^{y_l}(1-p_l)^{1-y_l}.
\end{aligned} \tag{3.8}$$

Clearly, by setting $\mathcal{P}(y'_l = 1) = \mathcal{P}(y_l = 1)$ for $l = 1, \ldots, 2^{h-1}$, the above equality holds. In other words, the conditional random vector $\mathbf{x}_{x_h=0} = \boldsymbol{G}' \otimes \mathbf{y}'$ for $\boldsymbol{G}' = \boldsymbol{G}_{:,1..2^{h-1}}$. ■

The above lemma establishes that the conditional random vector $\mathbf{x}_{x_h=0}$ can be represented as an OR mixing of 2^{h-1} independent components. Furthermore, the set

of independent components is the same as the first 2^{h-1} independent components of $\mathbf{x}$ (under proper ordering).

Consider a sub-matrix of data matrix $\boldsymbol{X}$, $\boldsymbol{X}^0_{(h-1)\times T}$, where the rows correspond to observations of $x_1, x_2, \ldots, x_{h-1}$ for $t = 1, 2, \ldots, T$ such that $x_{ht} = 0$. Define $\boldsymbol{X}_{(h-1)\times T}$, which consists of the first $h-1$ rows of $\boldsymbol{X}$. Suppose that we have computed the bICA for data matrices $\boldsymbol{X}^0_{(h-1)\times T}$ and $\boldsymbol{X}_{(h-1)\times T}$. From Lemma 4, we know that $\boldsymbol{X}^0_{(h-1)\times T}$ is a realization of OR mixtures of independent components, denoted by $\mathbf{y}^0_{2^{h-1}}$. Furthermore, $\mathcal{P}[\mathbf{y}^0_{2^{h-1}}(l) = 1] = \mathcal{P}(y_l = 1)$ for $l = 1, \ldots, 2^{h-1}$. Clearly, $\boldsymbol{X}_{(h-1)\times T}$ is a realization of OR mixtures of 2^{h-1} independent components, denoted by $\mathbf{y}_{2^{h-1}}$. Additionally, it is easy to see that the following holds:

$$\begin{aligned} &\mathcal{P}[\mathbf{y}_{2^{h-1}}(l) = 1] \\ &= 1 - [1 - \mathcal{P}(\mathbf{y}^0_{2^{h-1}}(l) = 1)][1 - \mathcal{P}(y_{l+2^{h-1}} = 1)] \\ &= 1 - (1 - p_l)(1 - p_{l+2^{h-1}}), \end{aligned} \tag{3.9}$$

where $l = 1, \ldots, 2^{h-1}$. Therefore,

$$\begin{aligned} p_l &= \mathcal{P}(\mathbf{y}^0_{2^{h-1}}(l) = 1), \qquad l = 1, \ldots, 2^{h-1}, \\ p_{l+2^{h-1}} &= 1 - \tfrac{1-\mathcal{P}(\mathbf{y}_{2^{h-1}}(l)=1)}{1-\mathcal{P}(\mathbf{y}^0_{2^{h-1}}(l)=1)}, \qquad l = 2, \ldots, 2^{h-1}, \\ p_{2^{h-1}+1} &= \tfrac{\mathcal{F}(x_h=1 \wedge x_i=0, \forall i \in [1 \ldots h-1])}{\prod_{l=1\ldots 2^h, l\neq 2^{h-1}-1}(1-p_l)}. \end{aligned} \tag{3.10}$$

The last equation above holds because realizations of $\mathbf{x}$ where ($x_k = 1$ while $x_i = 0; \forall i \in \{0, \ldots, k-1\}$) are generated from OR mixtures of $\mathbf{y}_{2^{k-1}}$'s only.

Let $\mathcal{F}(A)$ be the frequency of event A, we have the iterative inference algorithm as illustrated in Algorithm 1.

When the number of observation variables $m = 1$, there are only two possible unique sources, one that can be observed by x_1, denoted by [1]; and one that cannot,

Algorithm 1: Incremental binary ICA inference algorithm

`FindBICA` $(\boldsymbol{X})$
input : Data matrix $\boldsymbol{X}_{m\times T}$
init : $n = 2^m - 1$;
$p_h = 0, h = 1, \ldots, n$;
$\boldsymbol{G} = m \times (2^m - 1)$ matrix with columns corresponding all possible binary vectors of length m;
ε = the minimum threshold for p_h to be considered a real component;

1 **if** $m = 1$ **then**
2 $p_1 = \mathcal{F}(x_1 = 0)$;
3 $p_2 = \mathcal{F}(x_1 = 1)$;
else
4 **if** $\boldsymbol{X}^0_{(m-1)\times T} = \emptyset$ **then**
5 $p_{1\ldots 2^{m-1}} =$ `FindBICA` $(\boldsymbol{X}_{(m-1)\times T})$;
6 $p_{2^{m-1}+1} = 1$;
7 $p_{2^{m-1}+2\ldots 2^m} = 0$;
  else
8 $p_{1\ldots 2^{m-1}} =$ `FindBICA` $(\boldsymbol{X}^0_{(m-1)\times T})$;
9 $p'_{1\ldots 2^{m-1}} =$ `FindBICA` $(\boldsymbol{X}_{(m-1)\times T})$;
10 **for** $l = 2, \ldots, 2^{m-1}$ **do**
11 $p_{l+2^{m-1}} = 1 - \frac{1-p'_l}{1-p_l}$;
12 $p_{2^{m-1}+1} = \frac{\mathcal{F}(x_m=1 \wedge x_i=0, \forall i\in[1\ldots m-1])}{\prod_{l=1\ldots 2^m-1, l\neq 2^{m-1}+1}(1-p_l)}$;
13 **for** $h = 1, \ldots, 2^m$ **do**
14 **if** $(p_h < \varepsilon) \vee (p_h = 0)$ **then**
15 prune p_h and corresponding columns g_h;
16 **output**: p and $\boldsymbol{G}$

denoted by [0]. Their active probabilities can easily be calculated by counting the frequency of $(x_1 = 1)$ and $(x_1 = 0)$ (lines 1 – 3). If $m \geq 2$, we apply Lemma 4 and (3.10) to estimate p and $\boldsymbol{G}$ through a recursive process. $\boldsymbol{X}^0_{(m-1)\times T}$ is sampled from the columns of $\boldsymbol{X}$ that have $x_m = 0$. If $\boldsymbol{X}^0_{(m-1)\times T}$ is an empty set (which means $x_{mt} = 1, \forall t$) then we can associate x_m with a constantly active component and set the other components' probability accordingly (lines 4 – 7). If $\boldsymbol{X}^0_{(m-1)\times T}$ is non-empty, we invoke FindBICA on two sub-matrices $\boldsymbol{X}^0_{(m-1)\times T}$ and $\boldsymbol{X}_{(m-1)\times T}$ to determine $p_{1\ldots 2^{m-1}}$ and $p'_{1\ldots 2^{m-1}}$, then compute $p_{2^{m-1}+1\ldots 2^m}$ from (3.10) (lines 10 –

12). Finally, p_h and its corresponding column g_h in $\boldsymbol{G}$ are pruned in the final result if $p_h < \varepsilon$ (lines 13 – 15), where ε is a user-defined threshold.

Reducing computation complexity: Let $S(m)$ be the computation time for finding bICA given $\boldsymbol{X}_{m\times T}$. From Algorithm 1, we have,

$$S(m) = 2S(m-1) + c2^m,$$

where c is a constant. It is easy to verify that $S(m) = cm2^m$. Therefore, Algorithm 1 has exponential computation complexity with respect to m. This is clearly undesirable for large m's. However, we notice that in practice, correlations among x_i's exhibit locality, and the $\boldsymbol{G}$ matrix tends to be sparse. Instead of using a complete bipartite graph to represent $\boldsymbol{G}$, the degree of vertices in V (or the number of non-zero elements in $\boldsymbol{G}_{:,k}$) tend to be much less than m. In what follows, we discuss a few techniques to reduce the computation complexity by discovering and taking advantage of the sparsity of $\boldsymbol{G}$. We first establish a few technical lemmas.

Lemma 5 *If x_i and x_k are uncorrelated, then $\mathcal{P}(y_l = 1) = 0$, $\forall l$ s.t., $g_{il} = 1$ and $g_{kl} = 1$.*

Proof. We prove by contradiction. Suppose $\exists l$, s.t., $g_{il} = 1$, $g_{kl} = 1$, and $\mathcal{P}(y_l = 1) = 0$. Denote the l's by a set L. Let $u = \wedge_{l\in L} y_l$. From the assumption, u is non-degenerate. Without loss of generality, we can represent x_i and x_k as

$$x_i = u \vee v_1$$

$$x_k = u \vee v_2$$

where v_1 and v_2 are disjunctions of remaining non-overlapping components in x_i and x_k, respectively. From Lemma 3, we know that x_i and x_k are correlated. This contradicts the condition. ■

Lemma 6 *Consider the conditional random vector* $\mathbf{x}_{x_k=0} = [x_1, x_2, \ldots, x_{k-1}|x_k = 0]^T$ *from a set* $\mathbf{x} = [x_1, x_2, \ldots, x_{k-1}, x_k]^T$ *generated by the data model in* (3.16). *If* x_i *and* x_k *are uncorrelated,* $\mathbf{x}_{x_k=0}(i)$ *and* $\mathbf{x}_{x_k=0}(k)$ *are uncorrelated.*

Proof. This lemma is a direct consequence of Lemma 4 and Lemma 5. ■

Lemma 5 implies that pair-wise independence can be used to eliminate edges/columns in $\boldsymbol{G}$. Lemma 6 states that pair-wise independence remains true for conditional vectors. Therefore, we can treat the conditional vectors similarly as the original ones.

We also observe that for $\mathbf{x} = [x_1, x_2, \ldots, x_{k-1}, x_k]^T$ if (3.11) holds then there does not exist an independent component that generates x_k and some of x_j, $j = 1, 2, \ldots k-1$. In other words, x_k is generated by a "separate" independent component.

$$\mathcal{P}(x_1, x_2, \ldots, x_{k-1}, x_k) = \mathcal{P}(x_1, x_2, \ldots, x_{k-1})P(x_k) \tag{3.11}$$

Finally from (3.9), we see that $\mathcal{P}(y_{k-1}(l) = 1) \geq \max(p_l, p_{l+2^{k-1}})$. Note that $\mathcal{P}(y_{k-1}(l) = 1)$ is inferred from $\boldsymbol{X}_{(k-1)\times T}$, while the latter two are for $\boldsymbol{X}_{k\times T}$. This property allows us to prune the $\boldsymbol{G}$ matrix along with the iterative procedure (as opposed to at the very end).

Now we are in the position to outline our complexity reduction techniques.

T1 For every pair i and k, compute p-value of $\boldsymbol{X}_i$ and $\boldsymbol{X}_k$ Let the associated p-value be $p(i,k)$. The basic idea of p-value is to use the original paired data $(\boldsymbol{X}_i, \boldsymbol{X}_k)$, randomly redefining the pairs to create a new data set and compute the associated r-values. The p-value for the permutation is proportion of the r-values generated that are larger than that from the original data. If $p(i,k) > \epsilon$, where ϵ is a small value (e.g., 0.05), the corresponding columns in $\boldsymbol{G}$ and elements in $\mathbf{y}$ can be removed.

T2 We can determine the bICA for each sub-vector separately if the following holds,

$$\mathcal{P}(x_1, \ldots, x_k) = \mathcal{P}(x_1, \ldots, x_l)\mathcal{P}(x_{1+1}, \ldots, x_k).$$

T3 If the probability of the i'th component of $\boldsymbol{X}_{k\times T}$ $p_i < \epsilon$, then $\forall j$, s.t., $\boldsymbol{G}_{:,i} \prec \boldsymbol{G}_{:,j}$, the probability of the j'th component of $\boldsymbol{X}_{k'\times T}$ $p_j < \epsilon$ for $k' > k$. In other words, these columns and corresponding components can be eliminated.

From our evaluation study, we find the computation time is on the order of seconds for a problem size $m = 20$ on a regular desktop PC.

3.5 The inverse problem

Now we have the mixing matrix $\boldsymbol{G}_{m\times n}$ and the active probabilities $\mathcal{P}(\mathbf{y})$, given observation $\boldsymbol{X}_{m\times T}$, the inverse problem concerns inferring the realizations of the latent variables $\boldsymbol{Y}_{n\times T}$. Recall that n is the number of latent variables. Denote y_i to be the binary variable for the i'th latent variable. Let $\mathbf{x} = \boldsymbol{G} \otimes \mathbf{y}$. We assume

that the probability of observing $\boldsymbol{X}$ given $\mathbf{x}$ depends on their Hamming distance $d(\mathbf{x}, \boldsymbol{X}) = \sum_i |\boldsymbol{X}_i - x_i|$, and $\mathcal{P}(\mathbf{x}|\boldsymbol{X}) = p_e^{d(\mathbf{x},\boldsymbol{X})}(1-p_e)^{m-d(\mathbf{x},\boldsymbol{X})}$, where p_e is the error probability of the binary symmetric channel. To determine $\mathbf{y}$, we can maximize the posterior probability of $\mathbf{y}$ given $\boldsymbol{X}$ derived as follows,

$$\begin{aligned}
\mathcal{P}\{\mathbf{y}|\boldsymbol{X}\} &= \frac{\mathcal{P}\{\boldsymbol{X}|\mathbf{y}\}\mathcal{P}\{\mathbf{y}\}}{\mathcal{P}\{\boldsymbol{X}\}} \\
&= \frac{\mathcal{P}\{\boldsymbol{X}|\mathbf{y}\}\mathcal{P}\{\mathbf{y}\}}{\mathcal{P}\{\boldsymbol{X}\}} \\
&\overset{(a)}{=} \frac{\mathcal{P}\{\boldsymbol{X},\mathbf{x}|\mathbf{y}\}\mathcal{P}\{\mathbf{y}\}}{\mathcal{P}\{\boldsymbol{X}\}} \\
&\overset{(b)}{=} \frac{\mathcal{P}\{\boldsymbol{X}|\mathbf{x}\}\mathcal{P}\{\mathbf{y}\}}{\mathcal{P}\{\boldsymbol{X}\}} \\
&= \frac{\prod_{i=1}^m \mathcal{P}\{\boldsymbol{X}_i|x_i\}\prod_{j=1}^n \mathcal{P}\{y_i\}}{\mathcal{P}\{\boldsymbol{X}\}} \\
&= \frac{\prod_{i=1}^m p_e^{|x_i-\boldsymbol{X}|}(1-p_e)^{1-|x_i-\boldsymbol{X}|}\prod_{j=1}^n p_i^{y_i}(1-p_i)^{1-y_i}}{\mathcal{P}\{\boldsymbol{X}\}},
\end{aligned}$$

where $\mathbf{x} = \boldsymbol{G} \otimes \mathbf{y}$. (a) and (b) are due to the deterministic relationship between $\mathbf{x}$ and $\mathbf{y}$. Recall that $x_i = \bigvee_{j=1}^n (g_{ij} \wedge y_j), i = 1, \ldots, m$. With M devoting a "large enough" constant, we can use the 'big-M formulation [59] to relax the disjunctive set and convert the above relationship between $\mathbf{x}$ and $\mathbf{y}$ into the following two sets of conditions:

$$\begin{aligned}
x_i &\leq \textstyle\sum_{j=1}^n g_{ij}y_j, \ \forall i = 1, \ldots, m. \\
M \cdot x_i &\geq \textstyle\sum_{j=1}^n g_{ij}y_j, \ \forall i = 1, \ldots, m.
\end{aligned} \tag{3.12}$$

Here, since $\sum_{j=1}^n g_{ij}y_j \leq n$, we can set $M = n$. Finally, taking log on both sides and introducing additional auxiliary variable $\alpha_i = |\boldsymbol{X}_i - x_i|$, we have the the following

integer programming problem:

$$\begin{aligned}
\max_{\alpha,y}. \quad & \sum_{i=1}^{m} \left[\alpha_i \log p_e + (1-\alpha_i)\log(1-p_e)\right] \\
& + \textstyle\sum_{j=1}^{n} \left[(1-y_j)\log\left(1-p_j\right) + y_j \log p_j\right] \\
\text{s.t.} \quad & x_i \leq \sum_{j=1}^{n} g_{ij} y_j, \qquad \forall i = 1, \ldots, m, \\
& n \cdot x_i \geq \sum_{j=1}^{n} g_{ij} y_j, \qquad \forall i = 1, \ldots, m, \\
& \alpha_i \geq \boldsymbol{X}_i - x_i, \qquad \forall i = 1, \ldots, m, \\
& \alpha_i \geq x_i - \boldsymbol{X}_i, \qquad \forall i = 1, \ldots, m, \\
& \alpha_i, x_i, y_j = \{0, 1\}, \quad \forall i = 1, \ldots, m, j = 1, \ldots, n.
\end{aligned} \tag{3.13}$$

This optimization problem can be solved using ILP solvers. Note that p_e can be thought of the penalty for mismatches between x_i and $\boldsymbol{X}_i$'s.

Zero Error Case: If $\boldsymbol{X}$ is perfectly observed, containing no noise, we have $p_e = 0$ and $\alpha_i = \mathbf{x}_i - \boldsymbol{X}_i = 0$, or equivalently, $\mathbf{x}_i = \boldsymbol{X}_i$. The integer programming problem in (3.13) can be simplified as:

$$\begin{aligned}
\max_{y}. \quad & \sum_{j=1}^{n} \left[(1-y_j)\log\left(1-p_j\right) + y_j \log p_j\right] \\
\text{s.t.} \quad & \boldsymbol{X}_i \leq \sum_{j=1}^{n} g_{ij} y_j, \qquad \forall i = 1, \ldots, m, \\
& n \cdot \boldsymbol{X}_i \geq \sum_{j=1}^{n} g_{ij} y_j, \qquad \forall i = 1, \ldots, m, \\
& y_j = \{0, 1\}, \qquad \forall j = 1, \ldots, n.
\end{aligned} \tag{3.14}$$

Clearly, the computation complexity of the zero error case is lower compared to (3.13). It can also be used in the case where prior knowledge regarding the noise level is not available.

3.6 Applications in communication networks

In this section, we present some case studies on real-world application of bICA. In general, bICA can be applied to any problem that concerns identifying hidden source signals from binary OR mixtures. The proposed method therefore can find applications in many domains. In multi-assignment clustering [107], where boolean vectorial data can simultaneously belong to multiple clusters, the binary data can be modeled as the disjunction of the prototypes of all clusters the data item belongs to. In medical diagnosis, the symptoms of patients are explained as the result of diseases that are not themselves directly observable [37]. Multiple diseases can exhibit similar symptoms. In the Internet tomography [24], losses on end-to-end paths can be attributed to losses on different segments (e.g., edges) of the paths. In all above applications, the underlying data models can be viewed as disjunctions of binary independent components (e.g., membership of a cluster, presence of a disease, packet losses on a network edge, etc). In this dissertation, we only present some specific applications in communication networks and demonstrate how bICA can be effectively applied.

3.6.1 Optimal monitoring in multichannel wireless network

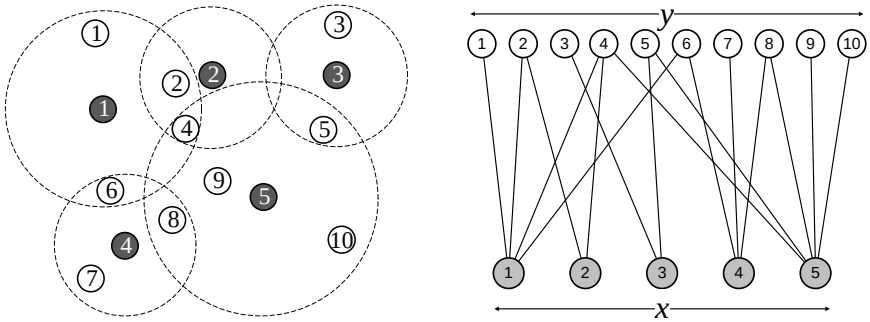
Passive monitoring is a technique where a dedicated set of hardware devices called *sniffers*, or monitors, are used to monitor activities in wireless networks. These devices capture transmissions of wireless devices or activities of interference sources in their vicinity, and store the information in trace files, which can be analyzed distributively

or at a central location. Most operational networks operate over multiple channels, while a single radio interface of a sniffer can only monitor one channel at a time. Thus, the important question is to decide the sniffer-channel assignment to maximize the total information (user transmitted packets) collected.

3.6.1.1 Network model and optimal monitoring

Consider a system of m sniffers, and n users, where each user u operates on one of K channels, $c(u) \in \mathcal{K} = \{1, \ldots, K\}$. The users can be wireless (mesh) routers, access points or mobile users. At any point in time, a sniffer can only monitor packet transmissions over **a single channel**. We represent the relationship between users and sniffers using an undirected bi-partite graph $G = (S, U, E)$, where S is the set of sniffer nodes and U is the set of users. An edge $e = (s, u)$ exists between sniffer $s \in S$ and user $u \in U$ if s can capture the transmission from u. If transmissions from a user cannot be captured by any sniffer, the user is excluded from G. For every vertex $v \in U \cup S$, we let $N(v)$ denote vertex v's neighbors in G. For users, their neighbors are sniffers, and vice versa. We will also refer to $\boldsymbol{G}$ as the binary $m \times n$ adjacency matrix of graph G. An example network with sniffers and users, the corresponding bipartite graph G, and its matrix representation $\boldsymbol{G}$ are given in Figure 3.2.

As mentioned earlier, the more complete information can be collected, the easier it is for a network administrator to make decisions regarding network troubleshooting. We can measure the quality of monitoring by the total expected number of active users monitored by the sniffers. Our problem now is to find an assignment of sniffers to channels so that the expected number of active users monitored is maximized. It



$$\boldsymbol{G} = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 \end{bmatrix} \begin{matrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{matrix}$$

$$\begin{matrix} y_1 & y_2 & y_3 & y_4 & y_5 & y_6 & y_7 & y_8 & y_9 & y_{10} \end{matrix}$$

Figure 3.2: A sample network scenario with number of sniffers $m = 5$, number of users $n = 10$, its bipartite graph transformation and its matrix representation. White circles represent independent users, black circles represent sniffers and dashed lines illustrate sniffers' coverage range.

can be casted as the following integer programming problem:

$$\begin{aligned} \max_{y,z} . \quad & \textstyle\sum_{u \in U} p_u y_u \\ \text{s.t.} \quad & \textstyle\sum_{k=1}^{K} z_{s,k} \leq 1, && \forall s \in S, \\ & y_u \leq \textstyle\sum_{s \in N(u)} z_{s,c(u)}, && \forall u \in U, \\ & y_u \leq 1, && \forall u \in U, \\ & y_u, z_{s,k} \in \{0,1\}, && \forall u, s, k, \end{aligned} \tag{3.15}$$

where the binary decision variable $z_{s,k} = 1$ if the sniffer is assigned to channel k; 0 otherwise. y_u is a binary variable indicating whether or not user u is monitored, and p_u is the active probability of user u.

3.6.1.2 Network topology inference with binary observations

From (3.15), it is clear that we need the network and user-level information in order to maximize the quality of monitoring. However, this information is not always available. We consider binary sniffers, or sniffers that can only capture **binary** information (*on* or *off*) regarding the channel activity. Examples of such kind of sniffers are energy detection sensors in spectrum sensing. Let $\mathbf{x} = [x_1, x_2, \ldots, x_m]^T$ be a vector of m binary random variables and $\boldsymbol{X}$ be the collection of T realizations of $\mathbf{x}$, where x_{it} denotes whether or not sniffer s_i captures communication activities in its associated channel at time slot t. Let $\mathbf{y} = [y1, y2, \ldots, y_n]^T$ be a vector of n binary random variables, where $y_j = 1$ if user u_j transmits in its associated channel, and $y_j = 0$ otherwise. Sniffer observations are thus disjunctive mixtures of user activities. The problem now is to infer the user-sniffer relationship (i.e. $\boldsymbol{G}$) and the active probability of users from the observation data (i.e. $\boldsymbol{X}$). In other words, given that the relationship between $\mathbf{x}$ and $\mathbf{y}$ follows $\mathbf{x} = \boldsymbol{G} \otimes \mathbf{y}$, we can use bICA to infer $\boldsymbol{G}$ and $\mathbf{y}$.

3.6.1.3 WiFi trace collection and evaluation results

We evaluate our proposed scheme by data traces collected from the University of Houston campus wireless network using 21 WiFi sniffers deployed in the Philip G. Hall. Over a period of 6 hours, between 12 p.m. and 6 p.m., each sniffer captured approximately 300,000 MAC frames. Altogether, 655 unique users are observed operating over three channels. The number of users observed on channels 1, 6,

11 are 382, 118, and 155, respectively. Most users are active less than 1% of the time except for a few heavy hitters. User-level information is removed leaving only binary channel observations from each sniffer. $\boldsymbol{G}$ and p are then inferred using bICA and input to the integer program (3.15) to find the best sniffer channel assignment that maximizes the expected number of active users monitored. Obviously, the more accurate the network model is inferred, the better the assignment is and the more users are monitored. We also vary the result by randomly select a subset of sniffers and observe the number of monitored users from this set of sniffers. Result is presented in Figure 3.3

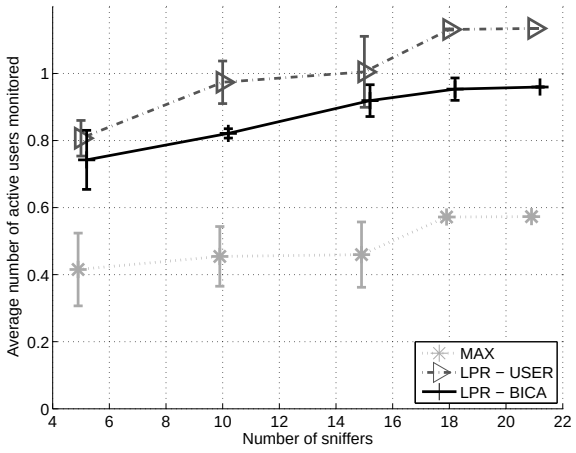


Figure 3.3: Expected number of active users monitored with the number of sniffers vary from 5 to 21.

In Figure 3.3, we compare the average of the number of active users monitored using the inferred $\hat{\boldsymbol{G}}$ and $\hat{p}$, and the ground truth. The integer programming problem (3.15) is solved using a random rounding procedure on its LP relaxation, which is shown to perform very close to the LP upper bound in our earlier work [31].

Note that most users are active less than 1% and the average active probability of users is 0.0014. The system consists of 655 unique users, therefore the average number of active users monitored is around 1 in each slot. For comparison, we also include a naive scheme (Max) that puts each sniffer to its busiest channel. Therefore, Max does not infer or utilize any structure information. From Figure 3.3, we observe that the sniffer-channel assignment scheme with bICA (LPR – BICA) performs close to the case when full information is available (LPR – USER), and much better than an agnostic scheme such as Max (MAX). This demonstrates that bICA can indeed recover useful structure information from the observations.

3.6.2 Primary user separation in cognitive radio networks

With tremendous growth in wireless services, the demand for radio spectrum has significantly increased. However, spectrum resources are scarce and most of them have been already licensed to existing operators. Recent studies have shown that despite claims of spectral scarcity, the actual licensed spectrum remains unoccupied for long periods of time [47]. Thus, cognitive radio (CR) systems have been proposed [88, 62, **?**] in order to efficiently exploit these spectral holes, in which licensed primary users (PUs) are not present. CRs or secondary users (SUs) are wireless devices that can intelligently monitor and adapt to their environment, hence, they are able to share the spectrum with the licensed PUs, operating when the PUs are idle.

One key challenge in CR systems is spectrum sensing, i.e., SUs attempt to learn

the environment and determine the presence and characteristics of PUs. Energy detection is one of the most commonly used method for spectrum sensing, where the detector computes the energy of the received signals and compares it to a certain threshold value to decide whether the PU signal is present or not. It has the advantage of short detection time but suffers from low accuracy compared to feature-based approaches such as cyclostationary detection [62, **?**]. From the prospective of a CR system, it is often insufficient to detect PU activities in a single SU's vicinity ("is there any PU near me?"). Rather, it is important to determine the identity of PUs ("who is there?") as well as the distribution of PUs in the field ("where are they?"). We call these issues the *PU separation problem*. Clearly, PU separation is a more challenging problem compared to node-level PU detection.

3.6.2.1 Solving PU separation problem with bICA

Consider a slotted system in which the transmission activities of n PUs are modeled as a set of independent binary variables $\mathbf{y}$ with active probabilities $\mathcal{P}(\mathbf{y})$. A *network interaction* is defined as the transmission from a PU that is sensed by a SU (monitor). The binary observations due to energy detection at the m monitor nodes are modeled as an m-dimension binary vector $\mathbf{x} = [x_1, x_2, \dots, x_m]^T$ with joint distribution $\mathcal{P}(\mathbf{x})$. It is assumed that presence of any active PU surrounding of a monitor leads to positive detection. If we let an (unknown) binary mixing matrix $\boldsymbol{G}$ represents the relationship between the observable variables in $\mathbf{x}$ and the latent binary variables in $\mathbf{y} = [y_1, y_2, \dots, y_n]^T$, then we can write $\mathbf{x} = \boldsymbol{G} \otimes \mathbf{y}$. The PU separation is therefore amenable to bICA. Given the estimated mixing matrix $\boldsymbol{G}$ and active probabilities

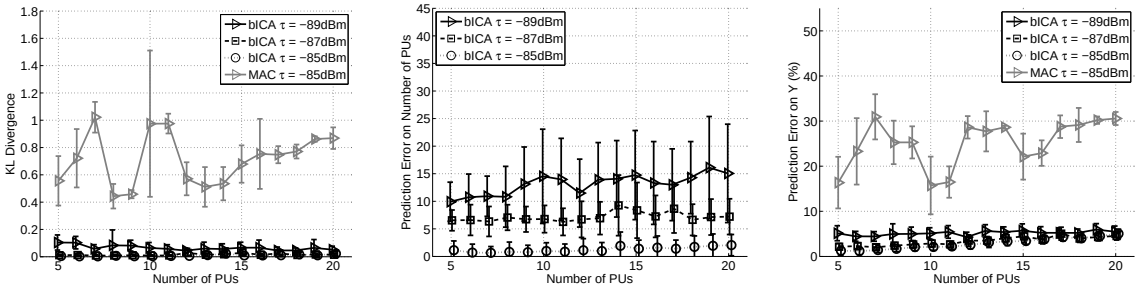
$\mathcal{P}(\mathbf{y})$, PU activity matrix $\boldsymbol{Y}$ can be inferred by solving the inverse problem 3.5.

3.6.2.2 Simulation setup and results

In the simulation, 10 SUs and n PUs are uniformly deployed in a 100x100 square meter area. Euclidean distances between PUs and SUs are computed from the coordinates of their locations. The PUs' transmit power levels P_t are fixed at 20mW; the noise floor N_0 at each SU is -95dBm; the center frequency f is 2.4GHz and the path loss exponent α is 3. The channel gain matrix is assumed to follow the standard Rayleigh fading model. The detection threshold τ for the SUs is set to -89dBm, -87dBm and -85dBm respectively, which corresponds to 6dB, 8dB and 10dB above the noise floor. The PUs' activities a modeled by a two-stage Markov chain with transition probabilities uniformly distributed over $(0, \frac{1}{30})$. We run the simulation for T slots and obtain the binary observations $\boldsymbol{X}$ and $\boldsymbol{X}_l$ for the binary OR and quantized linear mixture model, respectively. We compare performance of bICA and MAC [107] on two sets of experiments:

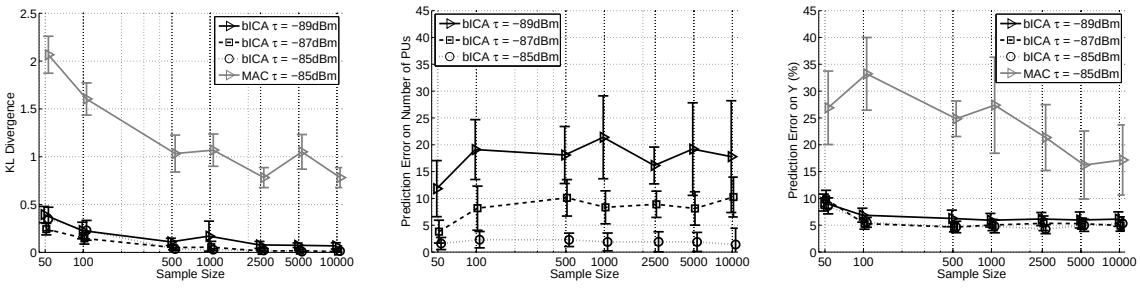
Varying the number of PUs: We fix the sample size $T = 10,000$ and vary the number of PUs from 5 to 20 to study the impact on the accuracy of the proposed method. Active probabilities of PUs are randomly distributed in $(0, 1)$ with mean = 0.1. Note that we need to specify the number of PU n as in input to MAC to speed up its execution. Therefore, no PU miscount is reported for MAC. We also omit results of MAC in noisy environments (τ = -89dBm and -87dBm) due to significantly inferior performance. Result is reported in Figure 3.4.

Varying the size of observations: In the second set of experiments, we fix the number of PUs $n = 10$ and study the impact of the observation size T. A small T (and thus insufficient observations) would lead to higher uncertainty while a large T incurs higher computation overhead. Furthermore, if T is too small, some PUs may never be active in the trace, making them impossible to be inferred. It is therefore interesting to investigate the effect of T on the accuracy and computation overhead of the proposed algorithm. Result is reported in Figure 3.5.



(a) PU transmission prob. error (b) Miscount of PUs (c) PU activity error ratio

Figure 3.4: Effects of the energy detection threshold and the number of PUs on inference results.



(a) PU transmission prob. error (b) Miscount of PUs (c) PU activity error ratio

Figure 3.5: Effects of the energy detection threshold and the size of observations on inference results. The x-axis is in logarithmic scale.

From the evaluation results, our proposed method outperforms MAC in every metric in all experiments. Not only does bICA achieve higher accuracy at all noise

levels, it also converges much faster, and thus significantly reduces the computation overhead. More detailed information on this problem can be found at [93].

3.6.3 Multicast tree topology inference

The problem of inferring logical tree topologies from end-to-end measurements can find applications in network tomography, where both the underlying multicast trees as well as losses on the logical links/nodes are not known. It was first studied by Cáceres *et al.* [20, 19]. Recently, Mao *et al.* [83] investigated topology inference and failure diagnosis for last mile cable networks. In a cable network, a single administrative area encompasses hundreds or thousands of end-customers (e.g., cable modems and VoIP phones) with thousands of intermediate distribution devices forming a tree topology. Intermediate devices are passive and/or too costly to monitor directly.

To see the connection between the disjunctive generation model in bICA and the tree topology inference problem, let us consider a simple example in Figure 3.6. Probes are injected from the root node and disseminated on the tree. Denote $y_l \in \{0, 1\}$, $l = 1, 2, \ldots, 8$ the random variables corresponding to the loss events (i.e., 1 for loss; 0 otherwise) on node l. Let $x_i = \{0, 1\}$, $i = 1, 2, \ldots, 5$ denotes whether a loss event is detected at the receiver i. Receivers themselves are assumed to be non-lossy. Clearly, x_i's follow a disjunctive relationship of the loss events on the nodes from the root to receiver i. For example, $x_1 = y_1 \vee y_2 \vee y_4$ since if any loss occurs at node 1, 2 or 4, receiver 1 will miss the probe packet. Probes packets transferred from one network node to another in this case is considered *network interactions*.

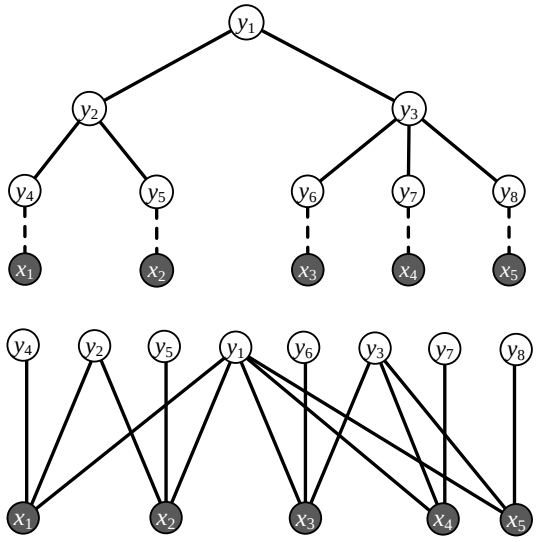


Figure 3.6: A canonical multicast tree and its bipartite graph transformation. In the bipartite graph, a receiver i is connected to a node j iff j is an ancestor of i.

3.6.3.1 Notation and network model

Let $\mathcal{T} = (V, L)$ denote the tree from a given source to many receivers. $\mathcal{T}$ consists of a set of nodes V and the set of links L. A link is an ordered pair $(j, k) \in V \times V$ denoting a link from node j to node k; a link is said to be internal if neither j nor k is the root node or a leaf node. We use $d(j)$ to denote the set of children of node j. (i.e., $d(j) = k \in V : (k, j) \in L$). For each node $j \in V$ excluding the root, there is a unique node $k = f(j)$ to be the parent of j, such that $(k, j) \in L$. We shall define $f^n(k)$ recursively by $f^n(k) = f(f^{n-1}(k))$. We further define $f^0(k) = k$. Node j is said to be an n-level descendant of k if $k = f^n(j)$ for some integer $n > 0$.

The root node $\in V$ is the source of the probes. Each leaf node in $R \subset V$ is attached a receiver (monitor). Without loss of generality, we assume there is no

loss on the receivers themselves. As a result, we can treat the receivers and the leaf nodes interchangeably. We model the loss of probe packets on $\mathcal{T}$ by a set of mutually independent Bernoulli processes. Each node $k \in V$ is associated with a Bernoulli process $\mathbf{y} = (y_k)_{k \in V}$ where each y_k takes a value in $\{0,1\}$. $y_k = 1$ corresponds to the event that a probe packet is lost on the node k, and 0 otherwise. Define $\mathbf{p} = \mathbb{P}(y = 1)$ to be the loss probability vector and $p_k \in \mathbf{p}, \forall k \in V$ to be the associated loss probability at each node k. For any node $k \in V \backslash R$, if the probe is lost at k (or $y_k = 1$), then it is lost at all children of k (and consequently at all descendants of k). $\boldsymbol{X}$ is an $m \times T$ collection of T realizations of vector $\mathbf{x}$ on the m receivers as T probes are dispatched from the root. For the t'th probe, $\boldsymbol{X}(r,t) = 0$ if it reaches the receiver r, and $\boldsymbol{X}(r,t) = 1$ if it was lost somewhere on the path from root node to r.

In this application, we can use an adjacency matrix $\boldsymbol{G} = [g_{ij}]_{m \times n}$ to represent $\mathcal{T}$, where n is the number of nodes ($n = |V|$) and m is the number of monitors or leaf nodes ($m = |R| < n$). $g_{ij} = 1$ if node j is an ancestor of the leaf node i, i.e., $j = f^h(i)$ for some integer $h \geq 0$; and $g_{ij} = 0$ otherwise. In other words, $g_{ij} = 1$ implies that probe packets originated at the root travel through node j before being received at leaf node i. If the packet is lost at node j (or $y_j = 1$), it will not be received at any receiver i that descends from node j ($x_i = 1, \forall i : g_{ij} = 1$). As an

example, the adjacency matrix for the multicast tree in Figure 3.6 is:

$$\boldsymbol{G} = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{matrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{matrix}$$

$$\begin{matrix} y_1 & y_2 & y_3 & y_4 & y_5 & y_6 & y_7 & y_8 \end{matrix}$$

3.6.3.2 The inference algorithm

Let $\boldsymbol{X}_{x_k=0}$ be the observation matrix, where $\boldsymbol{X}_{x_k=0}(j,t) \in \{0,1\}$, $\forall j \in R \backslash \{k\}$ and $x_k(t) = 0$. We proved that, when $x_k(t) = 0$, the binary random variables $x_j(t)$, $j \neq k$ can be grouped into clusters that are independent to each other (while dependency still exists among receivers within a same cluster). Additionally, receivers in each cluster $\mathcal{C}_h^k$ are descendants of a common ancestor $f^h(k)$ for some h. In the definition of the adjacency matrix $\boldsymbol{G}$, $g_{ij} = 1$ if node j is an ancestor of the leaf node i. Therefore, there exists an h such that $g_{ij} = 1, \forall i \in \mathcal{C}_h^k, j = f^h(k)$. This leads to the seqBICA algorithm, which inspects $\boldsymbol{X}_{x_k=0}$ for each $k \in \{1, 2, \ldots, m\}$. At each step, it partitions $R \backslash \{k\}$ into independent clusters and includes into $\boldsymbol{G}$ a new column (or equivalently a new independent component) for the new set of clusters generated. Our proposed approach is summarized in Algorithm 2.

Algorithm 2: Sequential bICA algorithm

`seqBICA`($\boldsymbol{X}$)
input : $\boldsymbol{X}$ = data matrix
init : m = number of receivers
T = number of observations
$n = 1$ (the tree is initialized with a root node)
$\boldsymbol{G}_{m\times 1} = [1, \ldots, 1]^{\mathrm{T}}$

for $i = 1, \ldots, m$ **do**
  $\boldsymbol{X}_{x_k=0} = \emptyset$
  for $j = 1, \ldots, T$ **do**
    if $\boldsymbol{X}(i,j) = 0$ **then**
      add the j'th column of $\boldsymbol{X}$ into $\boldsymbol{X}_{x_k=0}$
  $\mathcal{C} =$ `FindCluster`($\boldsymbol{X}_{x_k=0}$)
  foreach $U \subset \mathcal{C}$ **do**
    $col_{m\times 1} = [0, \ldots, 0]^{\mathrm{T}}$
    $col(j)_{\forall j \in U} = 1$
    if $col \notin \boldsymbol{G}$ **then**
      $\boldsymbol{G} = \boldsymbol{G} \cup col$
      $n = n + 1$

for $i = 1, \ldots, n$ **do**
  $R(i) = j : j \in \{1, 2, \ldots, m\} \wedge g_{ij} = 1$

$\Omega = \Lambda = \{1\}$
$\Phi = \{2, \ldots, n\}$
while $\Omega \neq \emptyset$ **do**
  for $j \in \Omega$ **do**
    $d(j) =$ `MinSetCover`($R(j), \bigcup_{l\in\Phi} \{R(l)\}$)
    $\Omega = \Omega \cup d(j)$
    $\Lambda = \Lambda \cup d(j)$
    $\Omega = \Omega \backslash j$
    $\Phi = \Phi \backslash \Omega$

Remove columns of $\boldsymbol{G} \notin \Lambda$

output Adjacency matrix $\boldsymbol{G}$
:

3.6.3.3 Evaluation

In this section, we first introduce performance metrics and then present performance of the proposed method on both synthetic and real traces. We compate the performance of seqBICA with that of DLT in [44]. We also run the original bICA algorithm and NetworkMD [83] on the same set of experiments, but omit the result due to inferior performance.

Performance metrics: We denote by $\hat{\boldsymbol{G}}$ and $\hat{\mathbf{p}}$ the inferred adjacency matrix and the inferred loss rate[1] of non-leaf nodes, respectively. Algorithm performance is evaluated using the following metrics:

- **Error on $\boldsymbol{G}$:** This metric indicates how accurately the adjacency matrix is estimated. It is defined by the Hamming distance between $\boldsymbol{G}$ and $\hat{\boldsymbol{G}}$ divided by its size. Denote $\boldsymbol{G}_{:i}$ to be the i'th column of $\boldsymbol{G}$ and d^H to be the Hamming distance between two vectors.

$$\bar{H}(\boldsymbol{G}, \hat{\boldsymbol{G}}) \triangleq \frac{1}{mn} \sum_{i=1}^{n} d^H(\boldsymbol{G}_{:i}, \hat{\boldsymbol{G}}_{:i}).$$

- **Error on p:** Prediction error in the inferred loss rate of independent internal nodes is measured by the root mean square error ratio between $\mathbf{p}$ and $\hat{\mathbf{p}}$, defined as below:

$$\bar{P}(\mathbf{p}, \hat{\mathbf{p}}) \triangleq \sqrt{\frac{\sum_{i=1}^{n} (\hat{p}_i - p_i)^2}{n}} \Big/ \frac{\sum_{i=1}^{n} p_i}{n}.$$

  Loss Rate Error Ratio can be interpreted as the fraction of the inferred loss probability that deviates from the true values.

[1] Having $\hat{\boldsymbol{G}}$, we calculate $\hat{\mathbf{p}}$ using the algorithm described in [20].

- **Error on $\mathcal{T}$:** This metric measures the percentage of time that the multicast tree topology is not correctly inferred, meaning $\hat{\boldsymbol{G}} \neq \boldsymbol{G}$. Clearly, the tree error rate is generally larger than the structure error ratio.

- **Computation time:** This metric measures how long an algorithm takes to finish inferring the network topology.

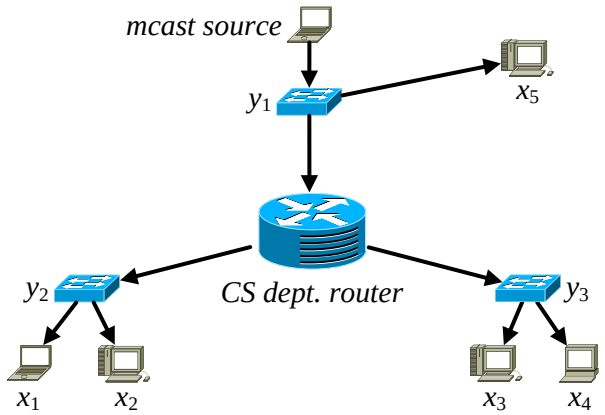
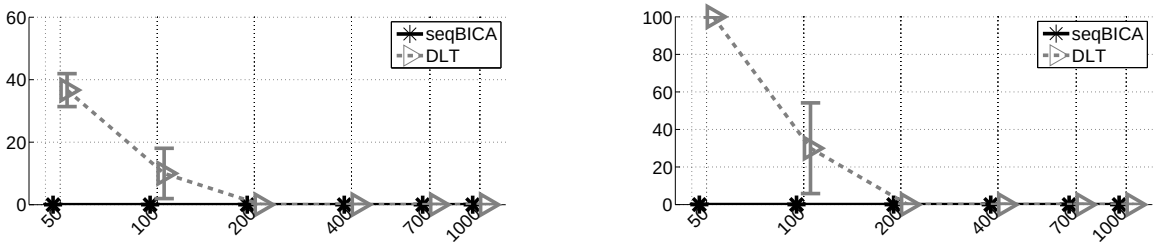


Figure 3.7: The multicast testbed with 6 computers and 3 switches. Arrows indicate the direction of the probe packets.

Virtual LAN experiments: We set up a small-scale multicast testbed at the CS department, University of Houston, which consists of 6 commodity workstations and 3 Netgear Gigabit GS105 switches as shown in Figure 3.7. The workstations run Debian and Windows OS. One computer acts as the multicast source and broadcasts UDP multicast messages, and the other 5 receivers listen to the multicast packets over Ethernet. We vary the number of observations T from 50 to 1,000, and collect the traces 10 times for each configuration. All experiments are conducted on a workstation with an Intel Core i5-750@2.66GHz processor and 6GB RAM. Algorithms are implemented in Matlab.

As shown in Figure 3.8, seqBICA significantly outperforms DLT when the number of probes are small. In fact, seqBICA can infer the topology correctly with 100% confidence interval even with 50 probe packets. DLT, on the other hand, converges much slower and requires at least 200 observations to correctly estimate the topology.



(a) Error on $\boldsymbol{G}$ (%) (b) Error on $\mathcal{T}$ (%)

Figure 3.8: Comparison between seqBICA and DLT. x-axis indicates the number of observations and y-axis indicates the error rate (%). Error bars are symmetric, and indicate standard deviation over 10 runs with different traces.

Simulated network experiments: We also have experimented with tree topologies used in related work as well as randomly generated trees with a good coverage of diverse scales and complexities. Simulation results are reported in Figure 3.9. In generating random tree topologies, the number of receivers varies from 10 to 20. Link loss probability on all trees is chosen uniformly in [0.05, 0.1]. p-value threshold ε in FindCluster, is set to 0.003. The results presented are the average of 50 runs for each scenario (with 50 different topologies for the random tree scenario).

From Figure 3.9(ii), we observe that when the number of observations T increases from 50 to 1,000, the structure error reduces drastically. When $T = 1{,}000$, structure error of both methods are close to 0. However, seqBICA converges much faster and have a lower error rate than DLT. The trend in Figure 3.9(iv) is similar, which shows

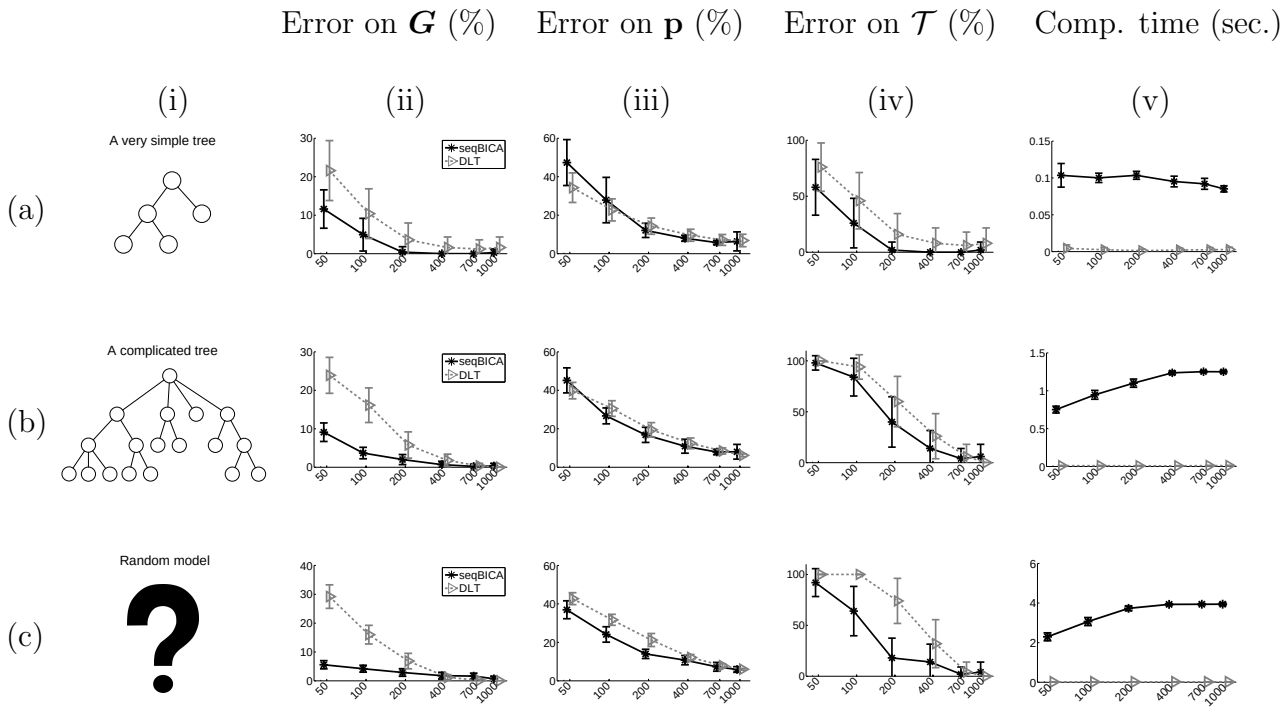


Figure 3.9: Experiment results with fixed and random tree topologies. The first column shows the topologies studied. The second to fifth columns show Structure Error Ratio, Loss Rate Error Ratio, Tree Error Rate and CPU runtime, respectively as T increase from 50 to 1,000. Each graph includes experiment results of seqBICA and DLT. Error bars are symmetric, and indicate standard deviation over 50 runs with different seeds.

a decrement in the tree error rate as the number of measurements increases for each algorithm. Again, seqBICA generally outperforms DLT in both convergence time and error rate. Performance of both algorithms in the $\mathbf{p}$ error test is comparable with seqBICA slightly outperforms the other in most cases. And finally, we see from Figure 3.9(v) that even though the computation time of seqBICA is negligible, DLT is still much faster.

3.6.4 Binary blind identification of wireless transmission technologies

The increasingly widespread adoption of wireless devices pose challenges in ensuring the satisfactory operations of wireless networks. Electro-magnetic interference (EMI) from unintended devices can potentially compromise the safety of mission-critical applications such as medical delivery. Even for non-critical applications such as VOIP or file transferring, excessive interference can greatly degrade their performance. Therefore, it is of great importance to monitor the spectrum usage in wireless networks. Efficient *data collection*, *representation*, *processing* and *mining* techniques need to be developed to facilitate real-time or offline diagnosis of wireless networks.

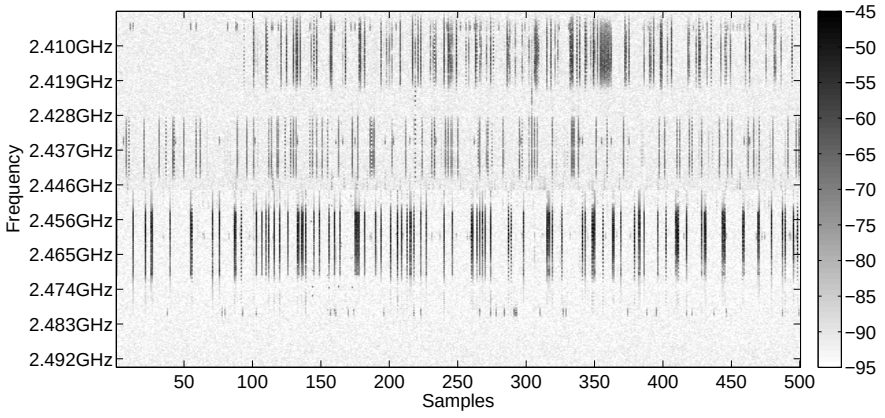


Figure 3.10: Spectrum measurement example

In this application, we make an initial but yet significant effort toward developing efficient techniques for spectrum monitoring. We make a case for the binary

representation of spectrum activities, and develop a novel technique for the identification of wireless transmission technologies from binary data. We term the proposed methodology "binary blind identification" ("blind" in a sense that no high-level features such as cyclostationarity are utilized). The identification is purely based on the spectrum bins that are occupied during transmissions of respective devices. Correlation in frequency domain is exploited such as one can distinguish whether the occupied spectrum bins are due to a single or multiple transmission technologies.

To get an intuition of the proposed approach, consider the spectrumography captured using a spectrum analyzer. Figure 3.10 shows the power spectrum density over time in gray scale. From Figure 3.10, we can clearly observe activates in the range 2.410GHz – 2.492GHz, which are likely to be associated with WiFi radio in channel 1, 6, 11 and Zigbee radios operating over channel 11, 17, 22, and 26. Even though the spectrum ranges 2.4603 – 2.4615GHz are occupied together by transmissions from both technologies at some times, it is not always the case, which allows us to separate the two technologies. The example demonstrates the promise of identifying wireless technologies based on spectrum activities. However, it is important to automate the above process and translate human knowledge into efficient algorithms since the amount of raw data is too large or too complicated to be processed manually.

3.6.4.1 Problem Formulation

Consider monitoring a wide-band spectrum divided into m sub-channels $c \in \mathcal{C} = \{c_1, c_2, \ldots, c_m\}$. Wireless devices using n different technologies $w \in \mathcal{W} = \{w_1, w_2, \ldots, w_n\}$ coexist in the same spectrum space of these m sub-channels. Each wireless device

is subject to one distinct technology, and each wireless technology in turn occupies a fixed set of continuous sub-channels. At any point in time, a transmission will result in power surge on its associated sub-channels. In this application domain, a we can consider a transmission that is captured by the spectrum sensing device an *interaction* on the network. We assume that activities of wireless devices operating on different frequency bands are independent. We represent the relationship between spectrum sub-channels and distinguished wireless technologies using an undirected bi-partite graph $\mathcal{G} = (\mathcal{C}, \mathcal{W}, \mathcal{E})$. An edge $e = (c, w) \in \mathcal{E}$ exists between the sub-channel $c \in \mathcal{C}$ and the technology $w \in \mathcal{W}$ if w operates on c. Channels not having any functional technology or technologies operating on channels outside the spectrum range of $\mathcal{C}$ will not be considered.

Let $\boldsymbol{G}$ be the $m \times n$ binary adjacency matrix of the graph $\mathcal{G}$. Entries in $\boldsymbol{G}$ encode the sub-channels occupancy status of each transmission technology. Using a spectrum sensing device such as the energy detector, we can observe the sub-channel occupancy status over time. Let the binary time series data representing spectrum occupancy be $\boldsymbol{X}_{m \times T}$, where T is the total of slots. $x_{ij} \in \boldsymbol{X}$ equals to 1 if sub-channel i is occupied at time j; and equals to 0 otherwise. Define $\mathbf{x}$ as the binary observation vector on all sub-channels; $x_{ij}, \forall i = 1, \ldots, m$ is therefore a realization of $\mathbf{x}$ at time slot j. Activities of wireless technologies are modeled by a set of mutually independent Bernoulli processes. Each technology $w \in \mathcal{W}$ is associated with a Bernoulli process $\mathbf{y} = (y_w)_{w \in \mathcal{W}}$, where each y_w takes a value in $\{0,1\}$. We call matrix $\boldsymbol{Y}_{n \times T}$ (that contains T realizations of vector $\mathbf{y}$) the activity matrix. $y_{wt} \in \boldsymbol{Y}$ equals to 1 corresponds to the event that at least one device using

technology w is transmitting at time t. Let $\mathbf{p} = \mathbb{P}(y_w = 1)$. Whenever a device using a particular technology is active, its associated sub-channels are occupied. Formally, the relationship between $\mathbf{x}$, $\boldsymbol{G}$, and $\mathbf{y}$ can be formulated as,

$$x_i = \bigvee_{j=1}^{n} (g_{ij} \wedge y_j), \ i = 1, \dots, m, \tag{3.16}$$

where $\wedge$ is Boolean *AND*, $\vee$ is Boolean *OR*, and g_{ij} is the entry in the i'th row and the j'th column of the unknown binary mixing matrix $\boldsymbol{G}$. For the ease of presentation, we introduce a short-hand notation for the above disjunctive model as,

$$\mathbf{x} = \boldsymbol{G} \otimes \mathbf{y}. \tag{3.17}$$

Therefore, from the binary encoded spectrum measurements matrix $\boldsymbol{X}$, we can apply bICA to infer $\boldsymbol{G}$, $\mathbf{p}$ as well as $\mathbf{y}$.

3.6.4.2 Evaluation

We denote by $\hat{\boldsymbol{G}}$, $\hat{\mathbf{p}}$, $\hat{\boldsymbol{Y}}$, and $\hat{n}$ the inferred mixing matrix, the inferred active probability vector, the inferred activity matrix, and inferred number of technologies, respectively. Four metrics are used to measure the accuracy of the inferred quantities.

- **Structure error ratio:** This metric is defined similar to error on $\boldsymbol{G}$ in Section 3.6.3.3
- **Transmission probability error:** This metric is defined similar to error on $\mathbf{p}$ in Section 3.6.3.3

- **Activity error ratio:** This metric measures how accurately the activity matrix is inferred by calculating the ratio of the absolute difference between $\boldsymbol{Y}$ and $\hat{\boldsymbol{Y}}$ over the size of $\boldsymbol{Y}$.

$$\bar{H}_y \triangleq \frac{1}{nT}\sum_{i=1}^{T} d(\boldsymbol{Y}_{:,i}, \hat{\boldsymbol{Y}}_{:,i}),$$

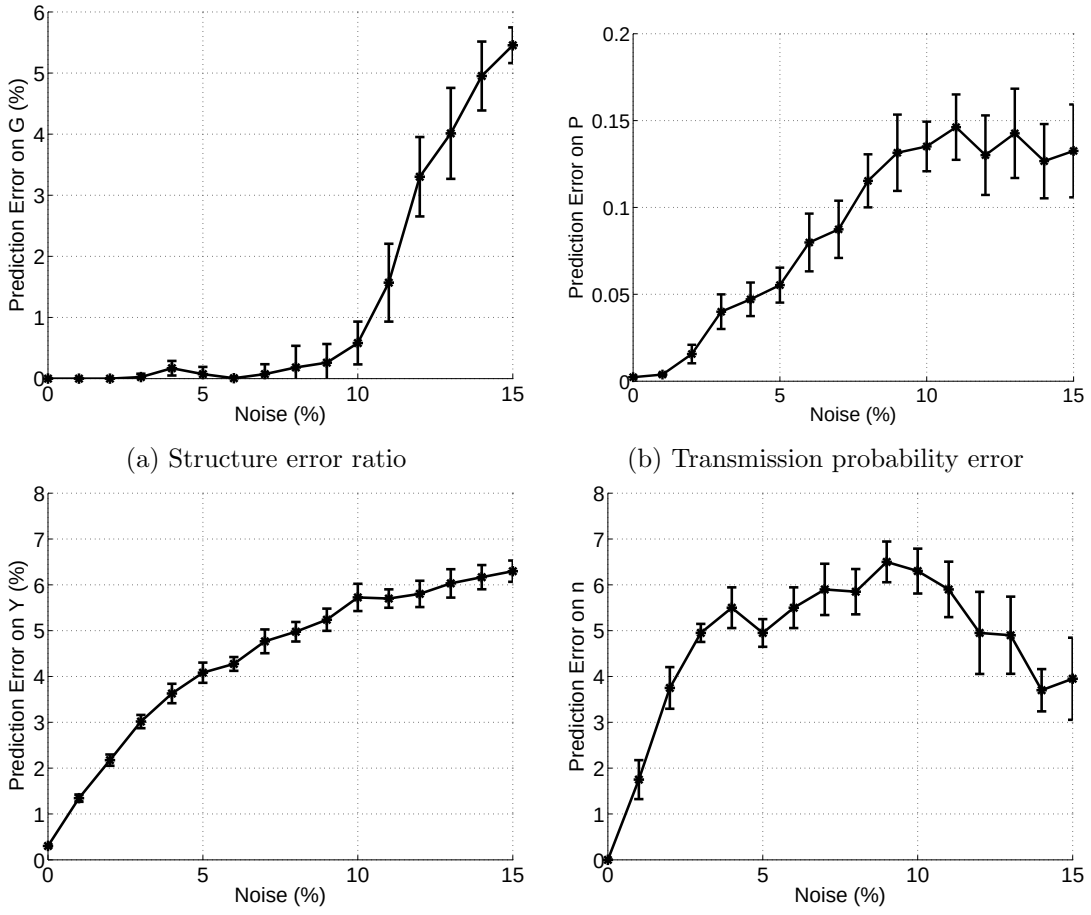
  where $\boldsymbol{Y}_{:,i}$ is the i'th column of $\boldsymbol{Y}$.

- **Technology quantity error:** We measure how many technologies are miscounted by calculating $|n - \hat{n}|$.

We consider a scenario in which multiple WiFi devices operate on 3 orthogonal WiFi channels 1, 6, 11 and ZigBee devices operate on 8 ZigBee channels 11 – 18. Thus, there are a total of 11 sets of occupied frequency bands (i.e., $n = 11$). Transmission probabilities of devices are randomly selected in [0.05, 0.1]. The sample size is T = 5,000. We divide the 2,400MHz – 2,499MHz spectrum into 100 sub-channels of 1MHz ($m = 100$). Spectrum sensing noise is introduced by randomly flipping the binary value in observation matrix $\boldsymbol{X}$ with error probability p_e varying from 0% to 15%.

Evaluation results presented in Figure 3.11 are the average value of 50 runs. From Figure 3.11 (a), we observe that the structure error ratio is negligible when $p_e \leq 10\%$. At higher noise levels ($p_e > 10\%$), structure error ratio monotonically increases. The transmission probability error and activity error ratio exhibits a similar trend as the error gets larger at higher noise (Figure 3.11 (b) and (c)). The proposed methodology shows its robustness against noise as the structure error ratio and activity error ratio are constantly kept below 7% even at very high noise level (15%). Figure 3.11 (d)

shows the number of inferred components matches the ground truth at zero noise and gets larger as noise increases. Surprisingly, at very high noise levels (10% – 15%), most noise components tend to have small active probabilities and are thus pruned, resulting a decrement in the quantity error.



(a) Structure error ratio

(b) Transmission probability error

(c) Activity error ratio

(d) Technology quantity error

Figure 3.11: Accuracy of the inference result with synthetic traces. Results are averages of 50 runs with different initial seeds with symmetric error bars indicating standard deviations.

3.7 Summary

In this chapter we provided a comprehensive study of the inference problems on communication networks. We identified a family of inference problems on networking domain where nodal interaction can characterized by a bipartite graph. We devised bICA algorithm and establish its key properties. Comparing to other similar approaches, bICA is not only faster, but also more accurate and robust against noise. We have also demonstrated the use of bICA in many network applications, including optimal monitoring in multichannel wireless network, primary user separation in cognitive radio network, multicast tree topology inference, and binary blind identification of wireless transmission technologies. We believe the methodology can be useful in many other application domains as well.

Chapter 4

Interactions on Online Social Networks

4.1 Introduction

A social network is defined as a network of interactions or relationships, where the nodes represent actors, and the edges represent the relationships or interactions between these actors. In general, the concept of social networks is not restricted to the specific case of an Internet-based social network such as Facebook. Conventional social network studies have the history before the advent of computers and the Internet. Milgram *et al.* [87] in 1967 hypothesized that any two random individuals on the planet are separated by at most six degrees of separation. This is referred to as the *small world* phenomenon. Extensive verification of such a hypothesis was not possible until many decades later, with the proliferation of online social networks.

Nowadays, the social network of interactions among a group of individuals plays a fundamental role in the spread of information, ideas, and influence. Such effects have been observed in many cases, when an idea or action gains sudden widespread popularity through *"word-of-mouth"* or *"viral marketing"* effects. To take a recent example from the technology domain, free e-mail services such as Microsoft's Hotmail, later Google's Gmail, and most recently Google's Google+ achieved wide usage largely through referrals, rather than direct advertising (Gmail and Google+ achieved wide usage at a time when the only way to obtain an account was through a referral). One also finds many examples in weblogs (blogs), where a piece of information spreads rapidly from one blogger to another before eventually being picked up by the mass media.

Analyzing and mining such large-scale complex networks are inherently difficult due to non-trivial features of the network itself. First, the network can be huge, comprising of billions of nodes and having one or two orders of magnitude more links. Any solution that targets such networks must be scalable and efficient to be able to produce meaningful results. Second, due to its nature, the graph might contain cycles, which make statistical inference problematic (since they are not fully characterized by simpler models such as Bayesian networks). Third, proposed algorithms should also take into account special network properties, like scale free and small diameter.

The main goal of our research is to understand the information diffusion process on social networks. How do the information and viruses spread over the network? What are the factors that affect the propagation process? Do they follow any kind

of pattern? How can we identify the set of most influential or critical nodes in the network? Answers to such questions are vital to a range of application areas, including viral marketing, biological network interactions, epidemiology and disease outbreak detection, etc.

We make several contributions to this topic:

- We crawl interaction traces from many Twitter communities and influence rank associated with each user. The dataset will be publicly available in a near future.

- We conduct an extensive analysis of Twitter and its power as a new medium for information propagation. Several interesting observations are drawn regarding the network properties and user behavior. A new information propagation model on Twitter is proposed to better characterize how the influence is spread.

- We introduce the budget influence maximization problem, a generalization of the traditional influence maximization (IM) problem. A greedy algorithm is proposed with a proven performance bound. Evaluations on both real-world social networks and randomly generated networks show its superior performance compared to state-of-the-art solutions. Furthermore, by varying properties of the synthetic networks, we are able to conduct study on the impact of graph structure on different IM algorithms.

4.2 Online social network datasets

4.2.1 Public datasets

Collecting social interaction datasets is essential to understand online social networks. To relieve researchers from the burden of individually collecting datasets, which typically requires considerable efforts, many online repositories provide public access to large network datasets. Some of the prominent repositories include:

Stanford Network Analysis Project (SNAP) [9]: A collection of more than 50 large network datasets from tens of thousands of nodes and edges to tens of millions of nodes and edges. It includes social networks, web graphs, road networks, internet networks, citation networks, collaboration networks, and communication networks.

Community Resource for Archiving Wireless Data At Dartmouth (CRAWDAD) [3]: A public archive aims to provide wireless trace data from many contributing locations. The staff also develop and provide tools for collecting, anonymizing, and analyzing the data.

Social Computing Data Repository [8]: A repository contains 19 datasets from many different social media sites, most of which have blogging capacity.

From the above sources, we pick several datasets which originate from a rich set of domains, including social, information, technological, and randomly generated networks. This diversity allows a more comprehensive study of network interactions and a thorough assessment of various network inference techniques. Details of the datasets are presented in Table 4.1.

Table 4.1: Online social datasets.

Name	Nodes	Edges	Description
soc-Epinions1	75,879	508,837	Who-trusts-whom network of Epinions.com
soc-LiveJournal1	4,847,571	6,8993,773	LiveJournal online social network
wiki-Vote	7,115	103,689	Wikipedia who-votes-on-whom network
email-EuAll	265,214	420,045	Email network from a EU research institution
cit-HepPh	34,546	421,578	Arxiv High Energy Physics paper citation network
cit-Patents	3,774,768	16,518,948	Citation network among US Patents
web-BerkStan	685,230	7,600,595	Web graph of Berkeley and Stanford
web-Google	875,713	5,105,039	Web graph from Google
amazon0302	262,111	1,234,877	Amazon product network from March 2 2003
amazon0312	400,727	3,200,440	Amazon product network from March 12 2003
amazon0505	410,236	3,356,824	Amazon product network from May 5 2003
amazon0601	403,394	3,387,388	Amazon product network from June 1 2003
soc-sign-epinions	131,828	841,372	Epinions signed social network

Even though many online datasets are available, they provide little insight in the dynamics of the network. We are not only interested in the network structure, but also need the interactions between them. Most existing social datasets only contain graphs with nodes representing users on the network and links representing the relationship. User identity is usually discarded from the dataset due to privacy concerns. Information exchanged between users (like tweets, messages), which is crucial to understand and analyze their relationship, is considered sensitive and cannot be published. Therefore we have to build a network crawler to collect new datasets from Twitter. By doing so, we can access not only the detailed information of each user but also their interactions. Furthermore, since we are also interested in users' social influence, we crawl data from influence measurement services to rank the user ability to drive action in online social networks.

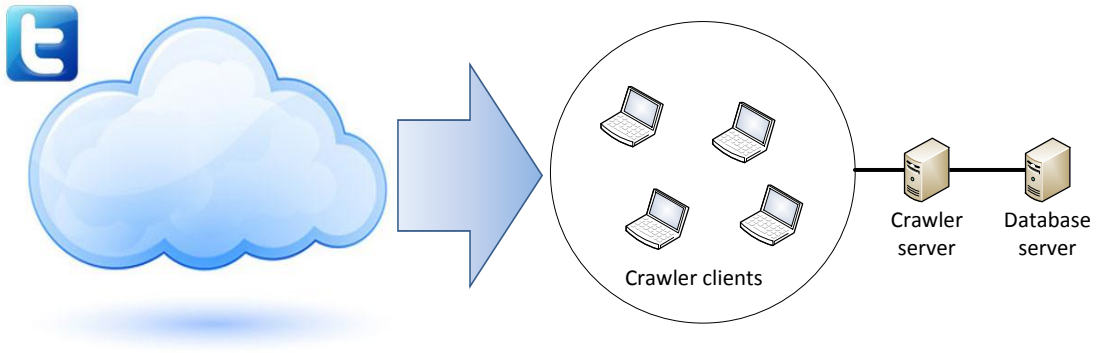


Figure 4.1: The Twitter crawler

4.2.2 Twitter crawler

4.2.2.1 Implementation

Twitter offers an Application Programming Interface (API) that is easy to crawl and collect data. We developed a network crawler in Java to extract data from Twitter and store in a MySQL database. However, due to the excessive amount of API requests that Twitter receives, they enforce a rate-limit of 350 requests per hour per IP address and terminate their whitelist program [11] (which allows a whitelisted IPs to make upto 20,000 requests per hour). This poses a challenge to the amount of attainable data since extracting the complete profile of a user would normally take upto 3 requests. To alleviate the above problem, we design a crawler following the client-server model as depicted in Figure 4.1.

Each crawler client with a different IP will make requests to crawl data from Twitter. Those data are aggregated at the crawler server. The server will then

check for data integrity and correctness before storing it in the database server. We control a PC pool with 50 machines making requests continuously from October to December 2012.

4.2.2.2 Data collection

The goal of the Twitter crawler is to obtain a complete dataset to capture nodes' relations and the interactions between them. Even with 50 machines crawling continously, obtaining a dataset on the scale of the entire Twitter sphere is not technically possible. Instead, we focus on crawling specific communities on Twitter where users share common interests on a trending topic.

Trending topics: Twitter tracks phrases, words, and hashtags that are often mentioned and classifies them as "trending topics". A hashtag is a convention among Twitter users to create and follow a thread of discussion by prefixing a word with a "#" character. By hashtagging the word, Twitter users create trends, that may draw the community's attention. By crawling the most popular hashtags on Twitter on many different topics [111], we obtain a diverse set of datasets that represent the most active communities on Twitter.

User profiles: Twitter profiles can be crawled from the list of user ids that participate in each trending topic. Twitter allows public access to a user's profile including name, location, web address, a short bio, and the number of tweets, unless the user set their profile to "private". The people who follow the user (followers) and those that the user follows (followees or friends) are also listed. Note that for the sake of

Table 4.2: Collected datasets from Twitter.

Hashtag	Nodes	Edges	Trend description
#android	172,817	1,695,021	Android phone, OS and applications
#at&t	74,200	426,518	Discussions on AT&T phone and service quality
#family guy	170,290	1,577,836	American animated TV show
#hiphop	93,440	1,862,110	Hip hop music genre
#iphone	94,928	501,295	Iphone and its applications
#ladygaga	19,525	65,158	American female singer
#marketing	226,606	19,123,496	General discussions on marketing and business
#nfl	55,200	703,090	American national football league
#sopa	36,993	474,173	U.S. bill to combat digital content piracy
#teaparty	19,772	3,169,181	American political party

graph compactness, we do not consider connections that are outside our observed communities. More specifically, we discard any connections and tweets of a twitterer to and from users not in our datasets.

To this end, we obtained 20.5 million user profiles, along with 420.2 million directed relations of follower and followee. We observe that 8.58% of the users set their profile to private, preventing us from accessing their information and relationships. We have to discard them from the dataset since the analysis could be biased. Incomplete datasets due to severely crawling rate and processing resource limit are discarded as well. The set of complete datasets are listed in Table 4.2.

Tweets: To collect tweets from a user, we first crawl the tweet history. Twitter keeps the history of most recent 3,200 tweets from a user. Older tweets are discarded. Since 3,200 tweets are insufficient to capture active user's history, we therefore monitor each user for a one-month period and capture all the tweets in that timing period. We collect the full text, the author, the time stamp, as well as the receiver if the

tweet is a reply. A total of 105 million tweets were collected.

4.2.2.3 Removing spam tweets

Spam tweets have increased in Twitter as the popularity of Twitter grows. Similar to spam webpages farms undermine the accuracy of PageRank on Google, spam tweets introduce noise and bias in our analysis. The Twitter Support Team suspends any user reported to be a spammer. Still, unreported spam tweets are present in our collected datasets, most notable in the #iphone and #android communities. We employ the well-known mechanism of the FireFox add-on, Clean Tweets [2]. Clean Tweets filters tweets from users who have been on Twitter for less than a day when presenting Twitter search results to FireFox. It also remove the tweets that contain three or more trending hashtags. We use the same mechanisms to remove spam tweets from our data.

4.2.3 Influence measurement services

With the proliferation of online social networks, the need for one to measure his social influence grows and many social influence scoring services emerge as a result. These services are measurement systems that assign the creators of online content a numerical rank based on behaviors that can trigger responses from other users. They promise to identify people who might serve as brand advocates (or "influencers") on social networks. Influence is measured by tracking and evaluating user actions from a diverse sites including Twitter, Facebook, Google+, Youtube, LinkedIn, etc. Among

many available services (such as Klout, PeerIndex, Kred, Empire Avenue, PROskore, etc.), we crawl the influence score from the two most popular ones:

Klout [4]: It is by far the most successful service. User influence score ranges from 1 to 100 with 100 being the most influential. U.S. President Barack Obama and pop star Justin Beiber are two persons that were scored 100. Klout measures influence mostly by using data points from Twitter, such as following count, follower count, retweets, list memberships, the influence of one's followers, etc.

PeerIndex [6]: PeerIndex also measure one's influence on a scale of 1 to 100. The service distincts itself by emphasizing its contributions at topic-by-topic level. The ability of users to drive conversations and provoke interactions are reported in different topics.

Although such services are mostly in beta phase and still constantly evolve, they somewhat reflect the influenceability of online users. Klout and PeerIndex score for all users in our database are crawled in one month from November to December 2012. We observe that 94.3% users have a Klout score, and 84.4% users have a PeerIndex score.

4.3 A quantitative study of Twitter and influence measurement services

4.3.1 Motivation

Recently, micro-blogging has emerged to be a new medium of communication. User can publish short messages (or statuses) to spread the information to his friends. Among the micro-blogging services, Twitter is the most notable one which claimed to have more than 500 million users by 2013 [68]. Basic functionalities of Twitter include disseminating *tweet* (a short message with a length limit of 140 characters) updating and socializing among users. A message can be *retweeted* by a recipient to further spread it beyond the reach of the original tweet's followers. Unlike other social network services that require users to grant permissions to other users to befriend them, Twitter employs a "free-to-follow" model which allows any user to follow and get update from others without seeking any permission. User A who follows B is called B's *follower*, while B is called A's *followee* or *friend* (we prefer the term *followee* to avoid confusion with other types of social friendship like that of Facebook, but will sometimes use the term *friend* interchangeably).

Being the most popular micro-blogging service, Twitter is also the most popular platform for social network research due to its open API that allows easy data access to researchers. Many investigations have been conducted to obtain a better understanding of the network's topological characteristics and user behaviors. Java *et al.* [66] conducted preliminary analysis on Twitter in 2007 with a small dataset of

76,000 users and 1 million tweets. The authors find user clusters based on user's interest to topics by clique percolation methods. Krishnamurthy *et al.* [75] analyzed the user characteristics by the relationships between the number of followers and that of followings. Zhao and Rosson [120] qualitatively investigated the motivation of using Twitter. Haewoon *et al.* [76] presented the first study on the entire Twitter sphere. Some interesting observations have been made, including non-power-law follower distribution, short effective diameter, and low reciprocity.

How to rank twitterers based on their influence is also an active research topic. A conventional indicator is the number of followers that one has. However, recent research [25, 114, 14] pointed out that this is not the case. Many researchers have strived to come up an intuitive and fair ranking system on Twitter. Kwak *et al.* [76], in an effort to indentify the most influential users on Twitter, applied several ranking metrics, including the number of followers, the number of retweets, and PageRank algorithm. The authors found that ranking results do not correlate, which means none of the above metrics is reliable. Cha *et al.* in [25] employed another metric: the number of times a user is mentioned. Weng *et al.* [114] proposed TwitterRank, an extension of PageRank algorithm to measure user influence.

Although a substantial body of work on Twitter analytic exists in literature, we approach the problem from a different angle. We present in this chapter an empirical study of Twitter in conjunction with influence measurement services, which are "social scoring systems" that assign each user a numeric score based on the user's ability to drive actions and provoke interactions within others. Such services are popular nowadays, including Klout, PeerIndex, Kred, Empire Avenue, PROskore,

just to name a few. The basic idea is as follow: those services scrape social network data, use it to create profiles of individuals, and assign each an "influence score". Twitter users do not have to register with the measurement services to have their profile evaluated, since their information can be obtained via Twitter's API interface. Once the user registers, the service will have full access to their data and provide more accurate measurement results. In exchange, user with high influence score (normally higher than 40) will be eligible for perks (discounted coupons, promotions, etc.) from many retailers. In this study, we analyze data from Klout and PeerIndex, as they are the two most popular services.

Rather than studing the entire Twitter network, for which a complete dataset can't practically be obtained, we focus on analyzing specific Twitter communities, identified by hashtags. This give us more information regarding how network properties and user behavior vary from one group of users to another. We present in this section results from the communities #ladygaga, #sopa, #android, and #marketing. Results from other communities are similar and therefore omitted.

From the analysis, we make several interesting observations, including non-power-law influence score distribution, asymmetric follower/followee distribution, strong reciprocity among users in a community, existence of homophily and hierarchial relationships on Twitter network. Furthermore, we find that whether or not a twitterer retweets a message is due to the influence from the first of his followees who posted that message. We will refer to this as the first influencer (FI) spreading model, which better characterize the spreading process on Twitter. We show through extensive evaluation that FI is more accurate in prediction influence spread on the Twitter

network.

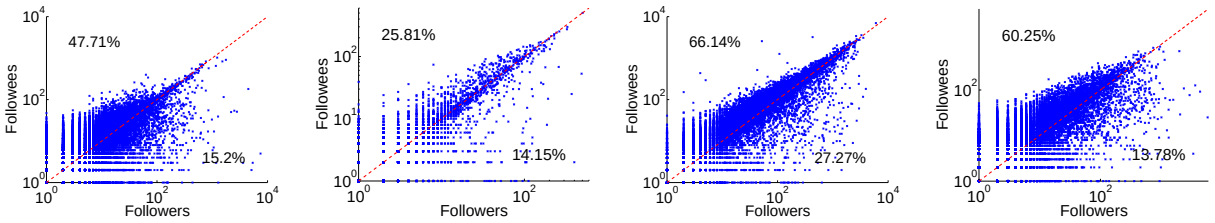
4.3.2 Reciprocity in following relationship

We begin our analysis by presenting the basic follower/followee distribution of our datasets in Figure 4.2. The number of followers and followees from each user is plotted in log scale. The main diagonal (dotted line) represents the perfect reciprocity where the number of followers is equal to the number of followees. Percentage above the diagonal represents the percentage of those who have more followees than followers. Unsurprisingly, more users are above the diagonal due to the "free-to-follow" mechanism of Twitter.

However, we find that there are a significant number of twitterers with equal number of followers and followews, most notable from #android and #ladygaga communities[1]. More specifically, this indicates a very strong reciprocity in these communities. To verify this finding, we first define a *mutual follower* to be a follower who is also a followee. Then we introduce a new metric, *reciprocal level*, defined as the ratio of the number of *mutual follower* to the number of followers. The histogram of *reciprocal level* on four communities is presented in Figure 4.3.

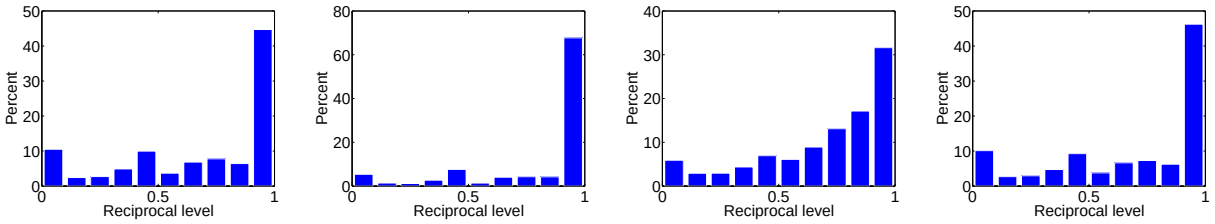
A large portion of users have reciprocal level 1, which means they follow all of those who follow them. Such a strong mutual relationship was not observed when the same study was conducted on the scale of Twitter [76]. Our results show that at a community level, users tend to have strong bidirectional connections to each other.

[1]Note that we only count relationships between users who are inside the community.



(a) #android (b) #ladygaga (c) #marketing (d) #sopa

Figure 4.2: Distribution of number of followers and followees.

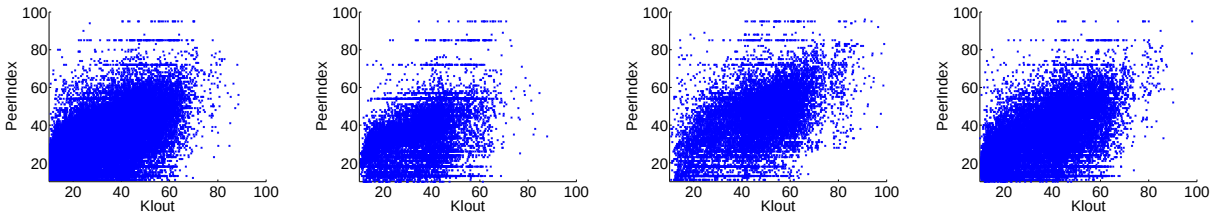


(a) #android (b) #ladygaga (c) #marketing (d) #sopa

Figure 4.3: Histogram of reciprocal level.

4.3.3 Distribution of influence scores

The influence score distributions from Klout and PeerIndex are presented in Figure 4.4. Note that a score of less than 10 is not an indicator of weak influenceability, but rather, like Klout and PeerIndex encounter issues in scraping the user's data for. This problem has been noted on Klout's developer blog [5]. As a result in our study, we discard users that have either score less than 10 (12.8% of the users). Since neither of the services publish their the ranking methodology, we can only infer that they user very different methods, as the correlation between the two is very low, on all datasets. Within the scope of this study, we suggest a simple metric to rank a user by taking the average of the two scores. We will refer to the new metric as the *digital influence* (DI) score.



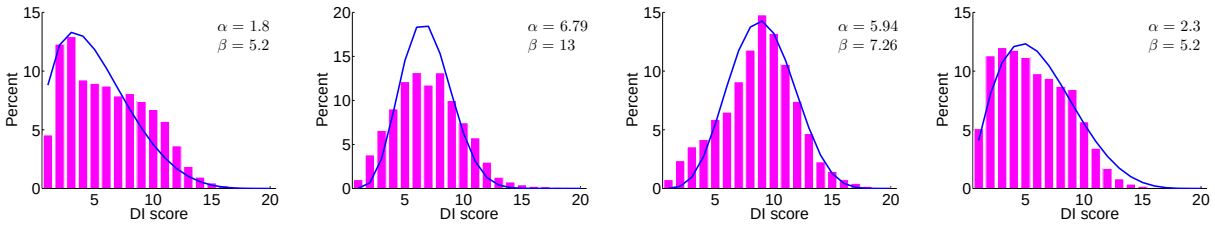
(a) #android (b) #ladygaga (c) #marketing (d) #sopa

Figure 4.4: Distribution of Klout and PeerIndex scores.

We observe that the DI score does not follow a power law distribution, but rather a beta distribution with two shape parameters $2 < \alpha < \beta$ and the mean value around 30 to 40. Figure 4.5 depicts the distribution of DI scores with its maximum likelihood estimates of the beta distribution. We also observe that the mean DI score varies from one community to another. As summarized in Table 4.3. The mean DI score is higher in the "#marketing" community since they include many business people and mass media entities, who have strong influence on others. On the contrary, the DI score is lower in the "#android" community. We observe that most tweets that contain an "#android" tag are from people who are playing games on Android. They tend to post tweets containing information of the game with an "#android" tag to receive perks or bonuses from the game provider. Those tweets will most of the time be discarded by other twitterers and therefore result a low average score in the "#android" community.

Table 4.3: Mean DI score in different communities.

Dataset	#android	#ladygaga	#marketing	#sopa
Mean DI score	29.15	34.33	45.04	33.25



(a) #android (b) #ladygaga (c) #marketing (d) #sopa

Figure 4.5: Histogram of DI score.

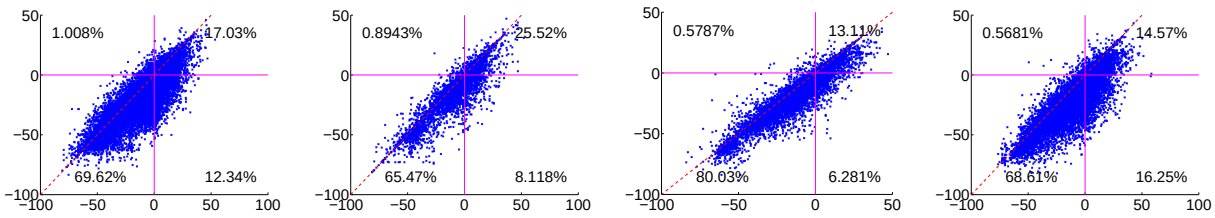
4.3.4 Hierarchy

Social hierarchy or stratification among humans is a well studied concept in sociology. Online social networks with their tremendous amount of available data give rise to new opportunities to study the social hierarchy for networks of different types and scales.

Although there is no official definition of stratification, recent studies show that hierarchy does exist on many online social networks, including Twitter. Researchers in [102, 82, 61] assume that people from connections in a social network based on their perceived social hierarchy. For instance, A following B means that B's rank is likely higher than A.

We conduct an analysis on our datasets to verify the hypothesis that the direction of social relationship encodes hierarchial information, from the influence ranking point of view. Let $N_{out}(u)$ and $N_{in}(u)$ be the set of node u's followers and followees, respectively. We define $\Delta_r(u)$ and $\Delta_e(u)$ to be the difference between the DI score of u and the mean DI score of u's followers and followees, respectively.

$$\Delta_r(u) = DI(u) - \frac{\sum_{\forall v \in N_{out}(u)} DI(v)}{|N_{out}(u)|} \text{ and } \Delta_e(u) = DI(u) - \frac{\sum_{\forall v \in N_{in}(u)} DI(v)}{|N_{in}(u)|}.$$



(a) #android (b) #ladygaga (c) #marketing (d) #sopa

Figure 4.6: Distribution of Δ_r and Δ_e. x-axis indicates Δ_r and y-axis indicates Δ_e.

We calculate $\Delta_r(u)$ and $\Delta_e(u)$ for all nodes in our datasets. Those that do not have any follower or followee will be discarded. Results from Figure 4.6 show that the majority of users have $\Delta_r > \Delta_e$, which means the mean score of their followees is higher than that of their followers. This proves the following relationship in Twitter contain hierarchial information where users tend to follow those who are more influential. Ideally, most users should have $\Delta_r > 0 > \Delta_e$. But in our result, the majority of them have $0 > \Delta_r > \Delta_e$, which could be attributed to several reasons. First, DI score follows the beta distribution with shaping parameters $\beta > \alpha$ as previously illustrated in Figure 4.5. Therefore, more users have DI score less than the mean score of their communities. Second, removal of users who do not have any follower or followee can also result the asymmetric Δ_r and Δ_e distribution.

4.3.5 Homophily

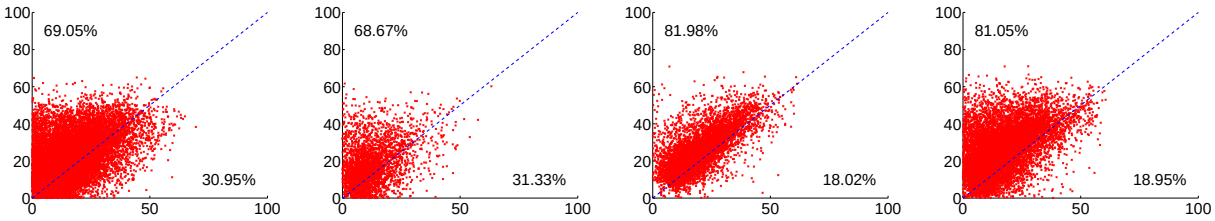
Homophily is a phenomenon where people's social networks "are homogenous with regard to many sociodemographic, behavioral, and interpersonal characteristics" [85]. In the context of Twitter, homophily implies that there are stronger connections between those who are "socially equal". Understanding homophily can help us build

better user models for personalization and recommender systems. Many previous studies [114] have verified homophily on Twitter along many dimensions, such as age, location, occupation, topical interest, expertise, etc. In this section, we study homophily from the perspective of user influenceability.

We test the hypothesis that users with similar influence are likely to be mutual followers. Let $N_{re}(u)$ be the set of reciprocal followers of u (those that are both u's follower and followee) and $N_{nre}(u)$ be the rest of u's followers who are not in $N_{re}(u)$. We have $N_{re}(u) = \{v|v \in N_{out}(u) \wedge u \in N_{out}\}$ and $N_{nre}(u) = \{v|v \in N_{out}(u) \wedge v \notin N_{re}(u)\}$. Define $\Delta_{re}(u)$ and $\Delta_{nre}(u)$ as average score distance of u to others in $N_{re}(u)$ and $N_{nre}(u)$, respectively.

$$\Delta_{re}(u) = \frac{\sum_{\forall v \in N_{re}(u)} |DI(u) - DI(v)|}{|N_{re}(u)|} \text{ and } \Delta_{nre}(u) = \frac{\sum_{\forall v \in N_{nre}(u)} |DI(u) - DI(v)|}{|N_{nre}(u)|}.$$

We calculate Δ_{re} and Δ_{nre} for all nodes in dataset and discard those that have empty N_{re} or N_{nre}. Homophily exists if mostly $\Delta_{re} < \Delta_{nre}$, indicating that nodes with similar influence will follow each other. Distributions of Δ_{re} and Δ_{nre} presented in Figure 4.7 show that the hypothesis is true on Twitter. We notice the portion of users that have $\Delta_{nre} > \Delta_{re}$ varies from one community to another, which indicates that some communities (#marketing, #sopa) have stronger homophily than others (#android, #ladygaga). This is probably because users have better awareness of their influence on some particular topics. Once influence is estimated by users, they will tend to befriend with those who have similar influence. There are many factors that decide the following relationship between users, our result shows that influenceability or a socially perceived version of it can be one of them.



(a) #android (b) #ladygaga (c) #marketing (d) #sopa

Figure 4.7: Distribution of Δ_{re} and Δ_{nre}. x-axis indicates Δ_{re} and y-axis indicates Δ_{nre}.

4.3.6 First-influencer diffusion model

4.3.6.1 Introduction

Diffusion models that explain how information is spread or how a product is adopted can generally be divided into two groups:

General threshold model [96]: each node u in the network has a threshold $\theta_u \in (0, 1]$, typically drawn from some probability distribution, and a monotone activation function $f_u : 2^V \rightarrow [0, 1]$. An edge from u to v is associated with a weight $w_{u,v}$. Nodes that are already active will spread the influence to their direct neighbors. Node u becomes active at time $t + 1$ if $f_u(S) \geq \theta_u$ where S is the set of active nodes at time t.

Cascade model [71]: when a node u first become active at time t, it has one chance to influence each inactive neighbor v with probability $p_{u,v}$. A successful attempt will cause v to become active at time $t + 1$. If multiple neighbors of v become active at time t, their activation attempts are sequenced in a random order, but assumed to happen within time slot t. The process terminates when there is no more newly activated node.

Among them, the independent cascade (IC) [54] model, where the influence probability $p_{u,v}$ is a constant, has been widely adopted in literature. Many researchers [106, 14, 86] apply the IC model to Twitter to solve the influence maximization problem or to study the spread of information. However, we observe that Twitter does not advocate such a spreading mechanism. When a twitterer u tweet a new message m, this message will be visible to his followers. In other words, u attempts to spread m to all of his followers. We say that m is spread from u to one of his followers v if v retweets m. The IC model assumes that if the first spread attempt fails, later attempts can still succeed with constant probabilities. This can be interpreted in the context of Twitter as follows: u can spread m to v with probability of success $p_{u,v}$. Assuming u fails, but m later spread to u' which is a followee of v, then m will have another chance to be retweeted by v with probability $p_{u',v}$. We notice that unlike other online social networks like Facebook, the current implementation of Twitter (both web and mobile platforms) does not promote to the user a same message again even if it is retweeted by one of his followees. In other words, v will not be aware of the fact that u' retweets m and thus, m has no chance to be retweeted by v if it fails in the first try.

This changes the way we understand influence spread on Twitter, and could have impact on many application areas, including viral marketing, Twitter trending and analyzing, etc. To capture such effect, we introduce the first influencer (FI) model. We will present the model formulation in the next section and experiment results in the subsequent section.

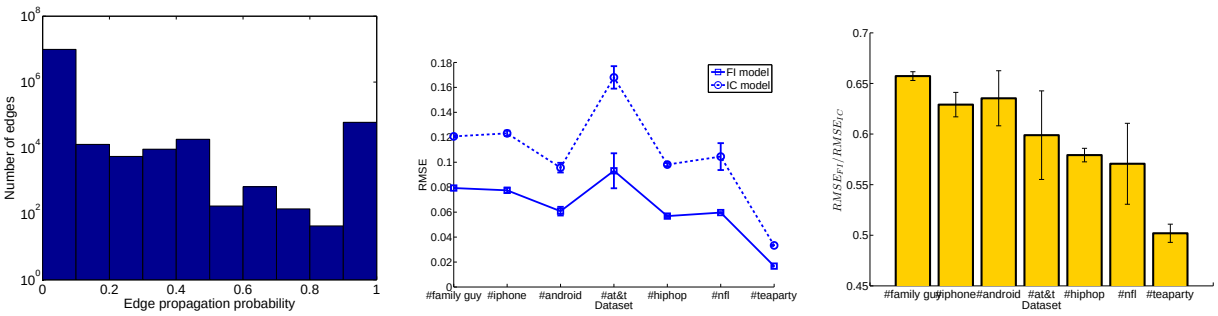
4.3.6.2 Formulation

Define the network $G = \{V, E\}$, with V and E are the sets of nodes and edges respectively. An active node u at time t will attempt to spread the information a to one of its inactive neighbors v with probability of success $p^a_{u,v}$, given that v is yet to be activated by any other nodes. If v is activated, it will in turn, try to activate its inactive neighbors in time $t + 1$. However, if v fails to be influenced at the first attempt, it will show strong resistance to similar attempts from it neighbors in the future and set $p^a_{u',v} = \varepsilon$ with any $u' \in N_{in}(v)$ and ε is a small number.

4.3.6.3 Experiments

In this section, we conduct experiments to compare our proposed FI model with the traditional IC model. Since there is no ground truth of the actual diffusion model, we conduct two sets of experiments on Twitter datasets to show the advantages of the proposed model. We will run experiments on a variety of communities #android, #at&t, #family guy, #hiphop, #iphone, #teaparty to see impact of the two models on different network structures and densities.

Model stability: In the first set of experiments, we assess the stability of each model. We first extract information cascades from the datasets using the algorithm in [56] and put them in a log, referred to as the *cascade log*. Then for each round of experiment, we randomly shuffle the *cascade log* and break it into 2 equal parts. A model is considered more stable should the parameters derived from the two datasets



(a) Histogram of inferred probabilities on 7 datasets

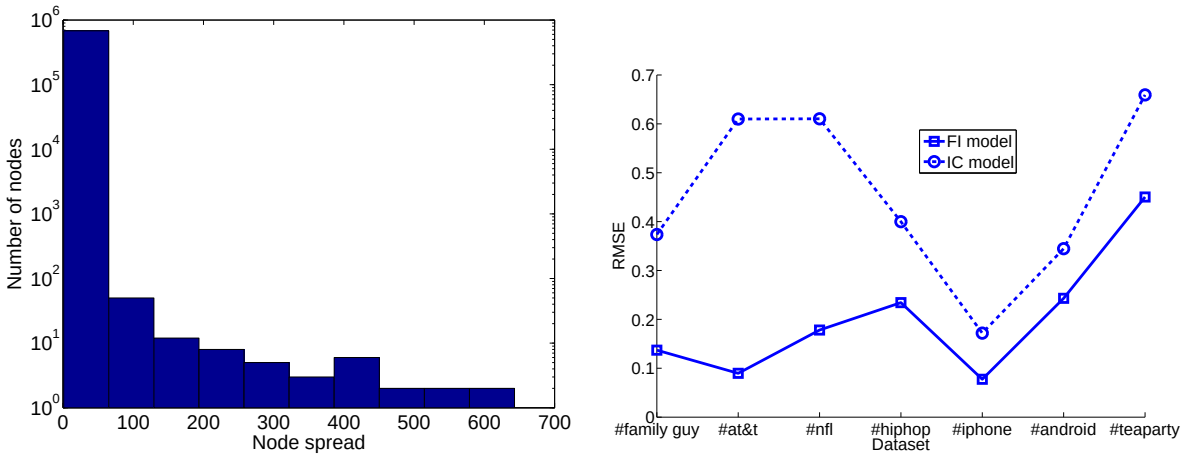
(b) $RMSE$ on 7 datasets

(c) $RMSE_{FI}/RMSE_{IC}$ ratio

Figure 4.8: Model stability comparison. Error bars indicate standard deviation.

are comparable. The parameters that we infer are the information propagation probabilities on the edges. By applying the algorithm described in [56] on both datasets, we calculate the probability vectors $\mathbf{p}_1, \mathbf{p}_2 = [p_{u,v}]_{1\times m}$ with any edge $(u, v) \in E$ and $m = |E|$ from the two datasets, respectively. Then we calculate the Root Mean Square Error [7] between $\mathbf{p}_1$ and $\mathbf{p}_2$, $RMSE(\mathbf{p}_1, \mathbf{p}_2) = \sqrt{\frac{\sum_{i=1}^{m}(p_1(i)-p_2(i))^2}{m}}$. Denote by $RMSE_{FI}$ and $RMSE_{IC}$ the value of $RMSE$ calculated from the FI and IC models respectively. The $RMSE$ indicates how deviated the two set of inferred parameters are. Higher $RMSE$ means the model is less stable. We conduct 100 rounds of experiments and report the average RMSE and standard deviation.

Figure 4.8(a) shows the aggregated distribution of inferred $\mathbf{p}_1$ and $\mathbf{p}_2$ on 7 datasets. From Figure 4.8(b), we see that both $RMSE_{FI}$ and $RMSE_{IC}$ are small in all datasets, which means both models perform well in practice. However $RMSE_{FI}$ is always smaller than $RMSE_{IC}$, showing the superior performance of the FI model. Figure 4.8(c) shows the ratio between $RMSE_{FI}$ and $RMSE_{IC}$, datasets are listed in the increasing density from left to right. On dense networks, the IC model tends



(a) Histogram of σ on 7 datasets

(b) Value of $RMSE$ on 7 datasets

Figure 4.9: Influence spread prediction comparison.

to overestimate the spread probabilities since each active neighbor has a chance to influence. As a result, we see the larger performance gap between FI and IC when the network is denser.

Influence spread prediction: The second set of experiments aim to determine which model is more accurate for influence spread inference. First we need to devise the propagation probability on edges. The process is similar to devising $\mathbf{p}_1$ and $\mathbf{p}_2$ above. However, this time we derive the propagation probability from the whole *cascade log* instead of splitting it into two parts. Denote by $\mathbf{p}_{FI}$ and $\mathbf{p}_{IC}$ the probability vectors determined from the FI and IC models, respectively. We compute the vectors of expected spread $\sigma_{FI}, \sigma_{IC} = [\sigma(u)]_{1\times n}$ from $\mathbf{p}_{FI}$ and $\mathbf{p}_{IC}$ for each node $u \in V$ and $n = |V|$ by running 10,000 rounds of Monte Carlo simulations (since exact calculation of σ from $\mathbf{p}$ is #P-complete [70]). To obtain the ground truth $\sigma(u)$, we calculate the average size of cascades from a node u.

Histogram of $\sigma(u)$ on 7 datasets is plotted in Figure 4.9(a). The majority of nodes have $\sigma = 1$ which means they can only influence themselves. The mean value of σ is 1.135. We also calculate the RMSE between the two distributions of σ_{FI} and σ_{IC} versus the ground truth σ and present the result in Figure 4.9(b). We see that the FI model outperforms the IC model in all datasets.

4.4 Budgeted influence maximization in online social networks

Given a fixed budget and an arbitrary cost for selecting each node, the budgeted influence maximization (BIM) problem concerns selecting a set of seed nodes to disseminate some information that maximizes the total number of nodes influenced (termed as *influence spread*) in social networks at a total cost no more than the budget. Our proposed seed selection algorithm for the BIM problem guarantees an approximation ratio of $(1-1/\sqrt{e})$. The seed selection algorithm needs to calculate the influence spread of candidate seed sets, which is known to be #P-hard. Identifying the linkage between the computation of marginal probabilities in Bayesian networks and the influence spread, we devise efficient heuristic algorithms for the latter problem. Experiments using both large-scale social networks and synthetically generated networks demonstrate superior performance of the proposed algorithm with moderate computation costs. Moreover, synthetic datasets allow us to vary the network parameters and gain important insights on the impact of graph structures on the performance of different algorithms.

4.4.1 Motivation

The social network of interactions among a group of individuals plays a fundamental role in the spread of information, ideas, and influence. Such effects have been observed in real life, when an idea or an action gains sudden widespread popularity through *"word-of-mouth"* or *"viral marketing"* effects. For example, free e-mail services such as Microsoft's Hotmail, later Google's Gmail, and most recently Google+ achieved wide usage largely through referrals, rather than direct advertising.

In viral marketing, one important question is given limited advertisement resources, which set of customers should be targeted such that the resulting influenced population is maximized. Consider a social network modeled as a graph with vertices representing individuals and edges representing connections or relationship between two individuals. The influence maximization (IM) problem tries to find a seed set S with cardinality $|S| = k$ in the graph such that the expected number of nodes influenced by S is maximized [40, 101, 70]. With the cardinality constraint, the submodularity nature of the influence spread renders a greedy algorithm with a $(1 - 1/e)$ approximate ratio that in each round picks the seed with the maximum influence spread and runs for k rounds. However the assumption of equal costs for all seed nodes seldom holds in practice. With the proliferation of influence score services such as Klout and PeerIndex, one can easily measure his influence in the social sphere and use that to negotiate the price for services he provides. The higher the influence score of a user, the more costly it is to persuade him.

We consider in this chapter a generalized version of the IM problem, namely,

the budgeted influence maximization (BIM) problem: given a fixed budget b and a random cost function c, to find a seed set S that fits the budget $\sum_{s_i \in S} c(s_i) \leq b$ and maximizes the number of influenced nodes. Clearly, BIM is more relevant in practice as there is typically a price associated with initializing the dissemination of information. With the budget constraint, we prove that a direct application of the simple greedy algorithm may result in unbounded performance gap.

In this chapter, we present a seed selection algorithm that can attain an approximation guarantee of $(1 - 1/\sqrt{e})$ (~ 0.394). One critical component of the seed selection process is the determination of influence spread of a set of seeds. Exact computation of influence spread is proven to be #P-complete [70]. Thus, efficient algorithms need to be devised. We first establish the linkage between influence spread computation and belief propagation on a Bayesian network (modeled as a directed acyclic graph (DAG)), where the marginal conditional dependency corresponds to the influence probabilities. Belief propagation has been extensively studied in literatures, and thus many exact or approximation algorithms can be leveraged to estimate the influence spread. For a general graph that contains loops, we propose two approximation algorithms that prune some edges in the graph to obtain a DAG that captures the bulk of influence spread. To reduce the number of candidate seed nodes, we localize the influence spread region such that in each round, only nodes that are affected by the previously selected seed need to be evaluated. Empirical study shows that the proposed algorithms can scale up to large-scale graphs with millions of edges with high accuracy. On real-world social network graphs, our methods achieve influence spread comparable to that by the greedy algorithm [70] and incur significant less

computation costs. In the unit-cost IM problem, the proposed methods outperform PMIA [28] in achievable influence spread at the expense of marginal increase in computation time. In the BIM problem, the proposed methods outperform CELF [78] in term of scalability and performance on dense graphs. We further study the effect of network structures on the performance of the algorithms.

Our main contributions in this topic can be summarized as follows:

- We propose a greedy algorithm for BIM with a constant approximation ratio.
- We cast the problem of influence spread computation on a DAG as an instance of belief propagation on a Bayesian network.
- We prove the #P-hardness of inference spread computation on a DAG.
- Two heuristics are proposed to construct DAGs from a general graph that capture the bulk of influence spread.
- We provide important insights on the impact of graph structures on performance of different algorithms.

4.4.2 Related work

Kempe *et al.* in [70] was the first to formulate the IM problem. The authors proved the submodularity of the influence spread function and suggested a greedy scheme (henceforth referred to as Greedy algorithm) with an incremental oracle that identifies, in each iteration, a new seed that maximizes the incremental spread. The

approach was proven to be a $(1 - 1/e)$-approximation of the IM problem. However, Greedy suffers from two sources of computational deficiency: 1) the need to evaluate many candidate nodes before selecting a new seed in each round, and 2) the calculation of the influence spread of any seed set relies on Monte-Carlo simulations. In an effort to improve Greedy, Leskovec *et al.* [78] recognized that not all remaining nodes need to be evaluated in each round and proposed the "Cost-Effective Lazy Forward" (CELF) scheme. Experimental results demonstrate that CELF optimization could achieve as much as 700-time speed-up in selecting seeds.

Chen *et al.* devised several heuristic algorithms for influence spread computation [29, 28, 30]. In Degree Discount [29], the expected number of additional vertices influenced by adding a node v in the seed set is estimated based on v's one-hop neighborhoods. It also assumes that the influence probability is identical on all edges. In [28] and [30], two approximation algorithms, PMIA and LDAG were proposed to compute the maximum influence set under the IC and LT models, respectively. In LDAG, it was proven that under the LT model, computing influence spread in a DAG has linear time complexity, and a heuristic on local DAG construction is provided to further reduce the compute time. We have proven in Section 4.4.4 that computing influence spread in a DAG under the IC model remains #P-hard. The marked difference between the two results arises from the fact that in the LT model, the activation of incoming edges is coupled so that in each instance, only one neighbor can influence the node of interest in an equivalent random graph model.

Literature on epidemiology is also related to the IM problem that identifies nodes

that can initiate viral propagation to most parts of the network. Defining the epidemic threshold of a network to be the condition that once satisfied, virus propagation on that network will dies out over time. The authors of [26] proved that the epidemic threshold for a network is exactly the inverse of the largest eigenvalue of its adjacency matrix. In a follow-up work [110], the authors used the previously defined epidemic threshold to quantify the vulnerability of a given network and devised a fast algorithm to choose the best k nodes to be immunized (removed) so as to minimize network vulnerability. [97] considered the immunization problem on dynamic networks. The key differences between work on viral immunization literatures and IM lie in the spreading model adopted (e.g.: SIS (susceptible-infected-susceptible) or SIR (susceptible-infected-recovered) vs. IC or LT) and whether the dynamics in the evolution of influence are of interest.

Most existing work on the IM problem only considers cardinality constraints. CELF [78] is the only applicable approach to the BIM problem. We will later show in evaluation results that the proposed methods outperform CELF in term of running time (several orders of magnitude faster) and performance on dense networks.

4.4.3 The budgeted influence maximization problem

4.4.3.1 Problem formulation

Consider a network modeled by a directed graph $\mathcal{G} = (V, E)$ with $|V| = n$ vertices and $|E| = m$ edges. For every edge $(u, v) \in E$, $p(u, v)$ denotes the probability of influence being propagated on the edge. In this section, we adopt the IC model [54]. Note

that we don't use the FI model proposed in Section 4.3.6 as this study is conducted on general social networks. Whether or not FI is applicable in social networks other than Twitter is still an open problem. Given a seed set $S \subseteq V$, the IC model works as follows. Let $S_t \subseteq V$ be the set of nodes that are (newly) activated at time t, with $S_0 = S$ and $S_t \cap S_{t-1} = \emptyset$. In round $t+1$, every node $u \in S_t$ tries to activate its neighbors in $v \in V \setminus \bigcup_{0 \leq i \leq t} S_i$ independently with probability $p(u, v)$. The influence spread of S, denoted by $\sigma(S)$, is the *expected* number of activated nodes given seed set S.

Kempe *et al.* [70] proved two important properties of the $\sigma(\cdot)$ function: 1) $\sigma(\cdot)$ is *submodular*, namely, $\sigma(S \cup \{v\}) - \sigma(S) \geq \sigma(T \cup \{v\}) - \sigma(T)$ for all $v \in V$ and all subsets S and T with $S \subseteq T \subseteq V$; 2) $\sigma(S)$ is *monotone*, i.e. $\sigma(S) \leq \sigma(T)$ for all set $S \leq T$. For any given spread function $\sigma(\cdot)$ that is both submodular and monotone, the problem of finding a set S of size k that maximizes $\sigma(S)$ can be approximated by a simple greedy approach.

Budgeted Influence Maximization: In BIM, each node u is associated with an arbitrary cost $c(u)$. The goal is to select a seed set $S \subseteq V$ such that the total cost of this set is less than a budget b. Denote by $c(S)$ the total cost of a set, i.e., $c(S) = \sum_{u \in S} c(u)$. Budgeted IM (BIM) can be formulated as an optimization problem:

$$\begin{aligned} \max_{S \subseteq V} \quad & \sigma(S) \\ \text{s.t.} \quad & c(S) \leq b \end{aligned} \tag{4.1}$$

When $c(u) \equiv 1, \forall u \in S$, BIM degenerates to the original IM problem. Thus,

Algorithm 3: Naive Greedy

input : $G = (V, E), b$

$S = \emptyset$
repeat
  $\delta(v) = (\sigma(S \cup v) - \sigma(S))/c(v), \forall v \in V$
  $u = \arg\max_{v \in V} \delta(v)$
  if $c(S \cup u) \leq b$ **then**
    $S = S \cup u$
  $V = V \backslash u$
until $V = \emptyset$;

output: S

we call IM the unit-cost BIM. Since IM is NP-hard, it is easy to see that BIM is NP-hard as well.

4.4.3.2 The seed selection algorithm

First, we consider an intuitive greedy strategy that selects at each step a node u that maximizes the incremental influence spread over cost ratio if the cost of u is less than the remaining budget. We hereby refer to this scheme as the *Naive Greedy* approach. Let r be the number of iterations executed and S_r be the seed set at step r. Note that $|S_r| \leq r$. At step $r + 1$, Naive Greedy calculates the incremental influence spread over cost ratio.

$$\delta(v) = (\sigma(S \cup v) - \sigma(S))/c(v), \forall v \in V \backslash S. \tag{4.2}$$

The algorithm chooses u if $u = \arg\max_{v \in V, c(s_r \cup v) \leq b} \delta(v)$. The algorithm terminates when no budget remains, or no node can be added to S. Naive Greedy is summarized in Algorithm 3.

Algorithm 4: Improved Greedy

input : $G = (V, E), b$

1 S_1 = result of Naive Greedy
2 $s_{max} = \arg\max_{v \in V} \sigma(v)$
3 $S = \arg\max(\sigma(S_1), \sigma(s_{max}))$

output: S

We first observe that Naive Greedy can have unbounded approximation ratio. Consider a network containing $l + 1$ nodes $V = \{u, v_1, v_2, \cdots, v_l\}$. Every pair in $v_1, v_2, \cdots, v_l$ is connected by an edge with influence probability one, while u is an isolated node. Let the cost $c(u) = 1 - \varepsilon$, $c(v_i) = l, \forall i = 1, \cdots, l$ and the budget $b = l$. The optimal solution will pick any node v_i and achieve an influence spread of l. In contrast, Naive Greedy picks u since it has the maximum incremental influence spread over cost ratio $1/1 - \epsilon > 1$. The resulting influence spread is 1. Thus, the approximation ratio for Naive Greedy is l.

Next, we show that Naive Greedy can be modified to achieve a constant approximation ratio. This algorithm is an adaptation of an algorithm first proposed by Khuller *et al.* [72]. We assume that there is no node with a cost greater than the budget b, as it will never be a feasible solution to BIM. Let S_1 be the seed set selected by Naive Greedy, we consider another candidate solution s_{max}, which is the node that has the largest influence. We compare the spread of S_1 and s_{max}, then output the one with higher influence spread. The process is summarized in Algorithm 4.

Theorem 2 *Algorithm 4 provides a* $(1-1/\sqrt{e})$*-approximation for the BIM problem.*

Proof. First we establish the following lemma. Let r be the number of iterations

executed by the repeat loop in Algorithm 3. Let S be the current seed set and S^* be the optimal seed set. Without loss of generality, we may renumber nodes that was added to S follow the chronicle order $S = \{u_1, u_2, \cdots, u_l\}$. Let $S_i = \bigcup_{j=1}^{i} u_j$ and let j_i be the index of the iteration in which u_i was considered.

Lemma 7 *After each iteration $j_i, i = 1, \cdots, l+1$, the following holds:*

$$\sigma(S_i) \geq \left[1 - \prod_{k=1}^{i} \left(1 - \frac{c(k)}{b}\right)\right] \sigma(S^*). \tag{4.3}$$

Proof. The proof of Lemma 7 was first presented by Khuller *et al.* in [72] for the budgeted maximum coverage problem, which is a special case of BIM where all the active edge probabilities are 1. Later, it was extended by Krause *et al.* (Lemma 3 in [74]) for general submodular functions. ■

Now we're in position to prove Theorem 2:

Proof. (Adapted from [72]) We prove Theorem 2 by case analyzing Algorithm 4.

- **Case 1:** If there exist at lease a node $u \in V$ which has spread greater than $\frac{1}{2}\sigma(S^*)$, then u or any other nodes which possess a greater spread, will be selected as the second candidate S_2. Algorithm 4 will therefore guarantee at least $\frac{1}{2}\sigma(S^*)$.

- **Case 2:** If there is no such node.

  - *Case 2.1:* If $c(S) < \frac{1}{2}b$, then we have $c(u) > \frac{1}{2}b, \forall u \notin S$ since there is no more node that can be added to S without violating the budget constrain.

W.l.o.g, we assume $S \neq S^*$. Therefore, $S^* \backslash S$ contains at most 1 node v, otherwise $c(S^*) > b$. By submodularity definition we have,

$$\begin{aligned} \sigma(S^* \cap S) + \sigma(v) &\geq \sigma((S^* \cap S) \cup v) + \sigma((S^* \cap S) \cap v) \\ &\geq \sigma(S^*) + \sigma(\emptyset) \\ &\geq \sigma(S^*). \end{aligned}$$

By assumption, we have $\sigma(v) < \frac{1}{2}\sigma(S^*)$, therefore $\sigma(S^* \cap S) \geq \frac{1}{2}\sigma(S^*)$. It follows that $\sigma(S) \geq \frac{1}{2}\sigma(S^*)$.

– *Case 2.2:* If $c(S) \geq \frac{1}{2}b$. We first observe that for $a_1, \cdots a_n \in \mathbb{R}$ and $\sum_{i=1}^{n} a_i \geq \alpha A$, the function,

$$\prod_{i=1}^{n} \left(1 - \frac{a_i}{A}\right)$$

is maximized when $a_i = \frac{\alpha A}{n}$. By Lemma 7, we have,

$$\begin{aligned} \sigma(S_i) &\geq \left[1 - \prod_{k=1}^{i} \left(1 - \frac{c(k)}{b}\right)\right] \sigma(S^*) \\ &\geq \left[1 - \left(1 - \frac{1}{2i}\right)^i\right] \sigma(S^*) \\ &\geq \left(1 - \frac{1}{\sqrt{e}}\right) \sigma(S^*). \end{aligned}$$

Thus, in the worst case, Algorithm 4 provides a $(1-1/\sqrt{e})$-approximation.

■

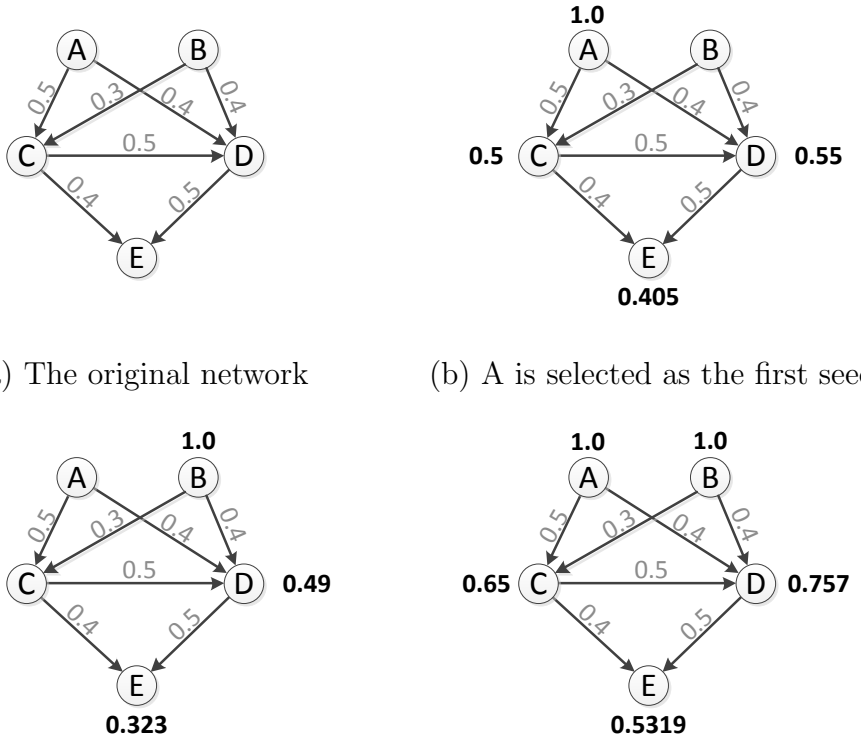
■

By considering the candidate solution with the maximum influence spread, Algorithm 4 guarantees the approximation ratio within a constant factor, while Algorithm

1 is unbounded. Note that Algorithm 4 is different from CELF proposed by Leskovec *et al.* in [78]. CELF runs Naive Greedy on the budgeted and the unit-cost (by setting all costs to one) versions of the problem, and selects the set with the maximum influence spread. While finding the seed set to maximize IM consumes more time than what it takes to select a single node with the largest spread, CELF can only guarantee a looser bound of $\frac{1}{2}(1 - 1/e)$ (~ 0.316).

Complexity: Let T be the maximum time needed to calculate the value of $\sigma(S), \forall S \subseteq V$. Algorithm 3 runs in $O(n^2T)$ time where n is the number of nodes (i.e. $n = |V|$). Finding S_1 costs $O(n^2T)$. s_{max} can be determined in in $O(nT)$ time. Therefore, Algorithm 4 runs in $O(n^2T)$ time. Note that in [72], Greedy with partial enumeration heuristic can achieve an approximation guarantee of $(1 - 1/e)$. However, the improvement is attained at the expense of much higher computation complexity of $O(n^4d)$ [27].

An Illustrative Example: Here we present an example to illustrate our seed selection process. Given a network with 5 nodes $V = \{A, B, C, D, E\}$ as depicted in Figure 4.10(a). Assuming we have the budget $B = 2$ and all nodal costs are 1, the problem becomes selecting two seed nodes to maximize network influence. In the first step, we have $S =$. The algorithm will first evaluate $\delta(v), \forall v \in V$. If A is selected as the first seed, it will be able to influence C, D, and E with probabilities of success



(a) The original network (b) A is selected as the first seed

(c) B is selected as the first seed (d) A and B are selected as seeds

Figure 4.10: An illustrative example of the seed selection algorithm. Edge propagation probabilities are in gray. Node active probabilities are in bold font.

0.5, 0.55, and 0.405, respectively (Figure 4.10(b)). Therefore, $\delta(A) = \sigma(A) = 2.455$. Similarly, we have $\delta(B) = 2.113, \delta(C) = 2.05, \delta(D) = 1.5$, and $\delta(E) = 1$. A will selected as the first seed since its incremental spread is largest. To select the second seed, we re-evaluate $\delta(v), \forall v \in V \backslash \{A\}$. If B is selected as the second seed, total spread on the network is increased to 3.9389 (Figure 4.10(d)), therefore $\delta(B) = 1.4839$. Similarly, we have $\delta(C) = 0.855, \delta(D) = 0.645$, and $\delta(E) = 0.595$. Having the largest incremental spread, B will be selected as the second seed. The algorithm

terminates since there is no more budget.

Algorithm 4 uses $\sigma(.)$ as a subroutine. The efficiency of $\sigma(.)$ computation is thus critical to the overall running time of the algorithm. In the following sections, we develop efficient algorithms for approximating the spread function $\sigma(.)$. We first consider the special case when the network is a directed acyclic graph (DAG). Then, we provide two DAG construction algorithms from a general network graph. Finally, some techniques to further optimize the execution of Algorithm 4 is presented.

4.4.4 Determining influence spread on DAGs

Given a seed set, estimating the value of $\sigma(.)$ from the seed set was proven to be a #P-complete problem [70]. We show in this section that under the IC model, the calculation of $\sigma(.)$ remains #P-complete even when the underlying network graph is a DAG. Then we establish the equivalence between computing $\sigma(.)$ on a DAG and the computation of marginal probabilities in a Bayesian network.

4.4.4.1 Hardness of computing influence spread on DAGs

In [70], Kempe *et al.* proposed an equivalent process of influence spread under the IC model, where at the initial stage, an edge (u, v) in $\mathcal{G}$ is declared to be *live* with probability $p(u, v)$ resulting in a subgraph of $\mathcal{G}$. A node u is active if and only if there is at least one path from some node in S to u consisting entirely of *live edges*. In general graphs, the influencer-influencee relationship may differ in one realization to another for bi-directed edges. In a DAG, on the other hand, such relationship is

fixed and is independent of the outcome of the coin flips at the initial stage (other than the fact that some of the edges may not be present). Let $x_u, u \in V$ denotes the binary random variable of the active state of node u, namely, $\mathbb{P}(x_u = 1) = p(u)$. For each node v in S, $\mathbb{P}(x_v = 1) = 1$. If a node $u \notin S$ does not have any parent in $\mathcal{G}$ then $\mathbb{P}(x_u = 1) = 0$. From $\mathcal{G}$, the conditional probability $p(x_u|x_{Par(u)})$ is uniquely determined by the edge probability, where $x_{Par(u)}$ denotes the states of the parents of node u. In other words, influence spread can be modeled by a Bayesian network. If node u does not have any parent, $p(x_u|x_{Par(x_u)}) = p(x_u)$. The joint distribution is thus given by,

$$p(x_1, x_2, \dots, x_n) = \prod_{i=1}^{n} p(x_i|x_{Par(x_i)}). \tag{4.4}$$

Given the outcome of coin flips C, $\sigma_C(S) = \sum_{u \in V} x_u$. Therefore,

$$\sigma(S) = \mathbb{E}(\sigma_C(S)) = \sum_{u \in V} \mathbb{E}(x_u) = \sum_{u \in V} p(u). \tag{4.5}$$

The second equality is due to the linearity of expectations. To compute $p(u)$, we can sum (4.4) over all possible configurations for $x_v, v \in V \backslash u$. Clearly, such a naive approach has complexity that is exponential in the network's treewidth. In fact, the marginalization problem is known to be #P-complete on a DAG. However, since computing influence spread on a DAG can be reduced to a special instance of the marginalization problem, it remains to be shown if the former problem is #P-complete. The main result is summarized in the following theorem.

Theorem 3 *Computing the influence spread $\sigma(S)$ on a DAG given a seed set S is #P-complete.*

Proof. The proof is an adaption of the proof in [28] and Valiant's original proofs of the #P-completeness of the s-t connectedness in a direct graph [112]. First, we define a few problems that are known or to be proven to be #P-complete.

Definition 1 *(SAT')*

Input: $F = c_1 \wedge c_2 \wedge \ldots c_r$, *where* $c_i = (y_{i1} \vee y_{i2})$ *and* $y_{ij} \in X$,

Output: $|\{(\mathbf{x}, \mathbf{t})|\mathbf{t} = (t_1, t_2, \ldots, t_n) \in \{1, 2\}^n$; *for* $1 \leq i \leq r$, $\mathbf{x}$ *make* $y_{i,k}$ *true for* $k = t_i$.

Definition 2 *(S-SET CONNECTEDNESS on DAG)*

Input: A DAG $\mathcal{D} = (V, E); s \in V; V' \in V$.

Output: Number of subgraphs of $\mathcal{D}$ *in which for each* $u \in V'$, *there is a (directed) path from* s *to* u.

Definition 3 *(S-T CONNECTEDNESS on DAG)*

Input: A DAG $\mathcal{D} = (V, E); s, t \in V$.

Output: Number of subgraphs of $\mathcal{D}$ *in which there is a directed path from* s *to* t.

To prove Theorem 3, we first establish the following lemma.

Lemma 8 $SAT' \preceq_p$ *S-T CONNECTEDNESS on DAG.*

Proof. Given F construct a DAG $\mathcal{D} = (V, E_1 \cup E_2)$ where $V = \{c_1, c_2, \ldots, c_{r+1}, x_1, \ldots, x_n, \bar{x}_1, \ldots, \bar{x}$
$E_1 = \{(x_i, c_j) | x_i$ appears in clause c_j in $F\} \bigcup \{(x_n, c_{r+1}), (\bar{x}_n, c_{r+1})\}$, and $E_2 = \{(x_i, x_{i+1}), (\bar{x}_i, x_{i+1})$
$i \leq n\} \bigcup \{(s, x_1), (s, \bar{x}_1)\}$. The direction of each edge follows the order of the pairs. $\mathcal{D}$ is a DAG as edges only go from x's of smaller index to larger ones, and from x's to c's. Note the $\mathcal{D}$ is multi-connected. The rest of the proof follows that in [112]. ■
Theorem 3 can then be proved using the same argument as in [28] with the exception that the reduction is from the S-T CONNECTEDNESS on DAGs due to Lemma 8. ■

4.4.4.2 Belief propagation

Belief propagation (BP) is a message passing algorithm for performing inference on graphical models, such as Bayesian networks and Markov random fields. It calculates the marginal distribution for each unobserved node, conditional on any observed nodes [117]. For *singly-connected* DAGs, where between any two vertices there is only one simple path, BP algorithm computes the exact solution with $O(n)$ complexity. For multi-connected DAGs, where multiple simple paths may exist between two vertices, belief propagation and many of its variants [117, 77, 90] have been shown to work well in general. Exact solutions such as junction tree [77] may incur the worst case complexity exponential to the number of vertices due to the need to enumerate all cliques in the DAG.

BP algorithms take as input a factor graph or a description of the underlying Bayesian Network. For each factor in the graph or a Bayesian node, a conditional probability table (CPT) is constructed. For a node v with the parent set $Par(v) =$

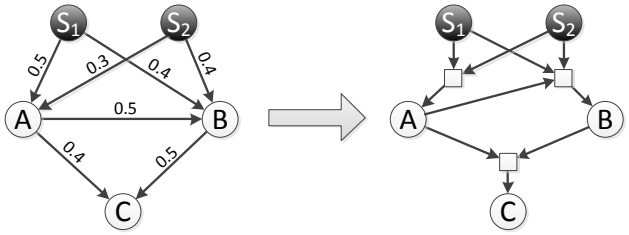


Figure 4.11: Converting a DAG into a factor graph.

		States of C	
A	B	0	1
0	0	1	0
0	1	0.5	0.5
1	0	0.6	0.4
1	1	0.3	0.7

Figure 4.12: CPT of C with two parents A, B

$\{par_1, par_2, \ldots, par_k\}$, its CPT consists of one column for each state and one row for each set of states its parents may assume. In the context of influence spread, each node only has two states: active (1) and inactive (0). Thus the number of rows in a CPT is 2^k. $\sigma(.)$ can then be determined by summing up the probability of nodes being active. An illustrative example of a factor graph and one of its CPT's is given in Figure 4.11 and 4.12.

Loopy belief propagation: The complexity of $\sigma(\cdot)$ calculation is dominated by the execution of the BP algorithm. A variety of BP algorithms exist. In this work, we adopt the Loopy Belief Propagation (LBP) algorithm, which has been shown to perform well for various problems [49, 84]. LBP takes $O(M^d)$ to estimate the active probability of a node, where M is the number of possible labels (states) for each

variable ($M = 2$), and d is the maximum in-degree. We denote by n_0 the number of vertices in a DAG. Thus, the complexity of LBP is $O(n_0 2^d)$.

Single-pass belief propagation: Calculating $\sigma(\cdot)$ with LBP produces highly accurate results, but the computation time remains to be high when the graph is multi-connected. The main complexity arises from the fact that the activation of the parents of a node may be correlated in a multi-connected graph. Thus, in computing the activation probability of the node, one needs to account for the joint distribution of its parent nodes. Next, we propose a single pass belief propagation (SPBP) algorithm that ignores such correlation in determining $\sigma(\cdot)$. Note that the heuristic is exact when the graph is singly-connected.

Let $\mathcal{D}(\cdot)$ be the input DAG. Consider a node $v \in \mathcal{D}(\cdot)$. Given the activation probabilities of its parents $Par(v)$, we approximate $p(v)$ as,

$$p(v) = 1 - \prod_{u \in Par(v)} (1 - p(u)p(u, v)). \tag{4.6}$$

The algorithm is summarized in Algorithm 5. It starts with the seed nodes and proceeds with the topological sorting order. The total complexity is $O(n_0 d)$. Clearly, SPBP is much faster than LBP.

4.4.5 DAG construction

In general, real social networks are not DAGs (with the exception of advisor-advisee and parent-child relationships, for instance, which exhibit a natural hierarchy). To apply the BP algorithm in computing influence spread, one needs to selectively prune

Algorithm 5: Single-Pass Belief Propagation (SPBP)

input : $\mathcal{D}(S)$

$\sigma(S) = 0$;
foreach $v \in \mathcal{D}(S)$ **do**
  if $v \in S$ **then**
    $p(v) = 1$
  else
    $p(v) = 1 - \prod_{u \in Par(v)}(1 - p(u)p(u, v))$
  $\sigma(S) = \sigma(S) + p(v)$

output: $\sigma(S)$

edges and reduce the graph to a DAG. Clearly, there are many ways to do so. The challenge is to find a DAG that approximates well the original graph in influence spread. In this section, we introduce two DAG construction algorithms, both retaining important edges where influences are likely to travel.

4.4.5.1 Localizing the influence spread region

One important observation in [28] is that the influence of a seed node diminishes quickly along a path away from the seed node. In other words, the "perimeter" of influence or the *influence region* of a seed node is in fact very small. One way to characterize the *influence region* of a node v is through the union of the maximum influence paths defined next.

Definition 4 *(Path Propagation Probability)*
For a given path $P(u, v) = \{u_1, u_2, \ldots, u_l\}$ *of length* l *from a vertex* u *to* v*, with* $u_1 = u, u_l = v$ *and* $u_2, \ldots, u_{l-1}$ *are intermediate vertices, define the propagation*

probability of the path, $p(P)$, as:

$$p(P(u,v)) = \prod_{i=1}^{l-1} p(u_1, u_{i+1}). \tag{4.7}$$

$p(P(u,v))$ can be thought as the probability that u will influence v if u is selected as a seed node.

Definition 5 *(Maximum Influence Path)*
Denote by $\mathcal{P}(\mathcal{G}, u, v)$ the set of all paths from u to v in $\mathcal{G}$. The maximum influence path $MIP(\mathcal{G}, u, v)$ from u to v is defined as:

$$MIP(\mathcal{G}, u, v) = \arg\max_{P} \{p(P) | P \in \mathcal{P}(\mathcal{G}, u, v)\}. \tag{4.8}$$

Ties are broken in a predetermined and consistent way such that $MIP(\mathcal{G}, u, v)$ is always unique, and every sub-path in $MIP(\mathcal{G}, u, v)$ from x to y is also in the $MIP(\mathcal{G}, x, y)$.

Definition 6 *(Maximum Influence Out-Arborescence)*
For a graph $\mathcal{G}$, an influence threshold θ, the maximum influence out-arborescence of a node $u \in V, MIOA(\mathcal{G}, u, \theta)$, is defined as:

$$MIOA(\mathcal{G}, u, \theta) = \bigcup_{v \in V, p(MIP(\mathcal{G},u,v)) \geq \theta} MIP(\mathcal{G}, u, v). \tag{4.9}$$

$MIOA(\mathcal{G}, u, \theta)$ is defined as the union of MIP's from u to all other nodes in V. MIP's with propagation probabilities less than a threshold θ are not included to reduce the size of $MIOA$. One can think of $MIOA(\mathcal{G}, u, \theta)$ as a *local region* where u can spread its influence to. $MIOA(\mathcal{G}, u, \theta)$ can be computed by first finding the Dijkstra tree rooted at u with edge weight $-\log(p(u,v))$ for edge (u,v), and then

removing the paths whose cumulative weights are too high. By tuning the parameter θ, influence regions of different sizes can be obtained. For a single node, its MIOA is clearly a tree. For multiple seed nodes, we build upon the idea of local influence region and propose two algorithms.

4.4.5.2 Building DAGs from a seed set

DAG model 1: We observe that any DAG has at least one topological ordering. Conversely, given a topological ordering, if only edges going from a node of low rank to one with high rank are allowed, the resulting graph is a DAG.

To obtain the topological ordering given seed set S, we first introduce a (virtual) super root node R that is connected to all seed nodes with edge probability 1. Let $\mathcal{G}_R = (V_{\mathcal{G}_R}, E_{\mathcal{G}_R})$ where $V_{\mathcal{G}_R} = V \cup \{R\}$ and $E_{\mathcal{G}_R} = E \cup \{(R, S_k) | \forall S_k \in S\}$. We build $MIOA(\mathcal{G}_R, R, \theta)$ by calculating a Dijkstra tree from R. After removing R and its edges from $MIOA(\mathcal{G}_R, R, \theta)$, we obtain a singly connected DAG $\mathcal{D}_1 = (V_{\mathcal{D}_1}, E_{\mathcal{D}_1})$ on which BP algorithms can be directly applied and used to estimate the influence spread from S. However, $\mathcal{D}_1(\cdot)$ is very sparse (with $n - k$ edges) since many edges are removed.

We then augment $\mathcal{D}_1(\cdot)$ with additional edges. Note that $MIOA(\mathcal{G}_R, R, \theta)$ provides a topology ordering. More specifically, let the rank of node v be the sum weight of the shortest path from R, namely,

$$r(v) = \min(-\log(p(P(s, v)))), \forall s \in S. \tag{4.10}$$

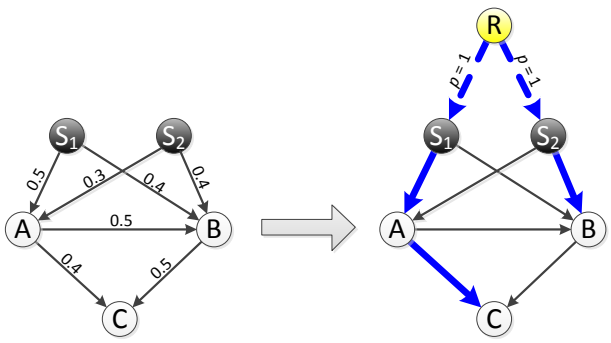
Algorithm 6: Calculate $\mathcal{D}_1(S)$ from a seed set S

input : $\mathcal{G}, S, \theta$

```
Build G_R = (V_{G_R}, E_{G_R})
D_1(S) = MIOA(G_R, R, θ)\R
Calculate r(v), ∀v ∈ V_{D_1} (Eq. (4.10))
foreach (u, v) ∈ V_{G_R} do
    if r(u) < r(v) and (u, v) ∈ E then
        D_1(S) = D_1(S) ∪ (u, v)
```

output: $\mathcal{D}_1(S)$

Rank grows as the node is further away from R. We include in $\mathcal{D}_1(\cdot)$ all edges in $\mathcal{G}$ whose end points are in $\mathcal{D}_1(\cdot)$ and go from a node with lower rank to one with higher rank. Clearly, the resulting graph is a DAG. The DAG constructing procedure is illustrated in Figure 4.13 and summarized in Algorithm 6.



Node	S_1	S_2	A	B	C
r(Node)	0	0	0.301	0.398	0.699

Figure 4.13: DAG due to Algorithm 6. S_1 and S_2 are seed nodes. Edges in $MIOA(\mathcal{G}_R, R, \theta)$ are in bold. (S_1, B), (S_2, A), (A, B), and (B, C) are added into $\mathcal{D}_1(S)$ to improve inference accuracy. $\theta = 0.0001$.

DAG model 2: In the second DAG model, we first compute the $MIOA$ from each seed node and take the union of $MIOA(\mathcal{G}, s, \theta), \forall s \in S$. Denote the resulting graph

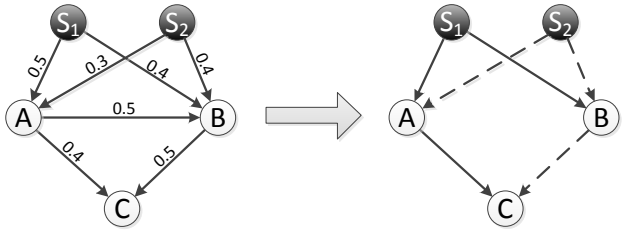
Algorithm 7: Calculate $\mathcal{D}_2(S)$ from a seed set S

input : $\mathcal{G}, S, MIOA(\mathcal{G}, v, \theta), \forall v \in V$

$\mathcal{D}_2(S) = \bigcup_{\forall s \in S} MIOA(\mathcal{G}, s, \theta)$
Calculate $r(v), \forall v \in V_{\mathcal{D}_2}$ (Eq. (4.10))
foreach $(u, v) \in \mathcal{D}_2(S)$ **do**
    if $r(u) \geq r(v)$ **then**
        $\mathcal{D}_2(S) = \mathcal{D}_2(S) \backslash (u, v)$

output: $\mathcal{D}_2(S)$

$\mathcal{D}_2(S) = (V_{\mathcal{D}_2}, E_{\mathcal{D}_2})$. Note that $\mathcal{D}_2(S)$ is not necessary a DAG as there could be cycles. To break the cycles, certain edges need to be removed. We adopt a similar approach as in Algorithm 6. A node v is associated with a rank $r(v)$ as in (4.10). Only edges that connect a lower ranked node to higher ranked node are retained. Clearly, the resulting graph is a DAG. The approach is summarized in Algorithm 7.



Node	S_1	S_2	A	B	C
r(Node)	0	0	0.301	0.398	0.699

Figure 4.14: DAG due to Algorithm 7. S_1 and S_2 are seed nodes. $\mathcal{D}_2(S)$ is the union of $MIOA(\mathcal{G}, S_1, \theta)$ (solid edges) and $MIOA(\mathcal{G}, S_2, \theta)$ (dashed edges). $\theta = 0.0001$.

The next proposition provides the relationship between DAGs constructed by Algorithm 6 and 7.

Proposition 1 *Given a fixed influence threshold θ, let $\mathcal{D}_1(\cdot) = (V_{\mathcal{D}_1}, E_{\mathcal{D}_1})$ and*

$\mathcal{D}_2(\cdot) = (V_{\mathcal{D}_2}, E_{\mathcal{D}_2})$ *be the DAGs constructed by Algorithm 6 and Algorithm 7. Then,* $V_{\mathcal{D}_1} = V_{\mathcal{D}_2}$ *and* $E_{\mathcal{D}_2} \subseteq E_{\mathcal{D}_1}$.

Proof. In both algorithms, a node v is not included in the DAG if and only if $r(v) > \theta$. Thus, $V_{\mathcal{D}_1} = V_{\mathcal{D}_2}$.

To show $E_{\mathcal{D}_2} \subseteq E_{\mathcal{D}_1}$, it suffices to show that $\forall (u, v) \in E_{\mathcal{D}_2}$, $(u, v) \in E_{\mathcal{D}_1}$. Since $(u, v) \in E_{\mathcal{D}_2}$, $(u, v) \in E$ and $r(u) \leq r(v)$. Therefore, according to Algorithm 2, $(u, v) \in E_{\mathcal{D}_2}$. Clearly, the converse is not true as some edges in $E_{\mathcal{D}_1}$ may not be part of the $MIOA$ from any seed node. ■

Computation complexity: Building the Dijkstra tree from a source node takes $O(n_0 \log n_0)$, where n_0 is the maximum number of vertices in the resulting DAG. Calculating the node rank $r(\cdot)$ takes $O(n_0)$. The union operation in DAG 2 takes $O(n_0 - 1)$ and the edge augmenting and pruning in DAG 1 and DAG 2 takes $O(m_0)$ and $O(\min(m_0, k(n_0 - 1)))$, respectively, where m_0 is the maximum number of edges in a DAG and k is the seed set cardinality.

Therefore, the running time of DAG 1 and DAG 2 are $O(n_0 \log n_0)$ and $O(n_0)$, respectively. Note that DAG 2 calculation requires the availability of $MIOA(\mathcal{G}, v, \theta), \forall v \in V$ first, which can be built at the initialization stage at the cost of $O(nn_0 \log n_0)$. Assuming that k is small and θ is properly selected, we have $n_0 \ll n$.

4.4.6 Acceleration of seed selection algorithm

In each round of Naive Greedy, a seed node with the maximum incremental spread-cost ratio is selected, namely, $v = \max_{v \in V \setminus S} \delta(v)$. Recall that $\delta(v) = (\sigma(S \cup v) - \sigma(S))/c(v)$ is the spread increment ratio of v under S. Initially, when $S = \emptyset$, $\delta(v) = \sigma(v)/c(v)$. Evaluating $\delta(v)$ at each iteration for all $v \in V$ dominates the overall computation complexity.

To accelerate the execution of Naive Greedy, one can try to improve on two aspects, namely, 1) limiting the candidate set of nodes to pick from for the next seed, and 2) reducing the complexity of computing the spread increments. CELF algorithm [78] eliminates many nodes from being evaluated. We focus on the second aspect. The proposed mechanism can be used in conjunction with the idea from CELF.

Recall in Section 4.4.5.1, we use $MIOA$ to localize the influence region of a node. Consider for now that influence from a node can only reach nodes in its $MIOA$. Then, we make the following claim.

Proposition 2 *Given the current seed set S, adding u to S will not change the spread increment of v, namely, $\delta_S(v) = \delta_{S \cup u}(v)$ if $MIOA(\mathcal{G}, u, \theta)$ and $MIOA(\mathcal{G}, v, \theta)$ have no common vertex.*

Proof. It is easy to see that by limiting the spread from u in $MIOA(\mathcal{G}, u, \theta)$, then $p(w), \forall w \in MIOA(\mathcal{G}, v, \theta)$ will not be affected by the inclusion of u in the seed set. ■

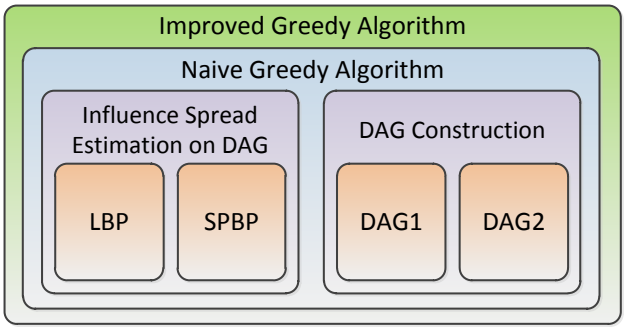


Figure 4.15: The building blocks of our proposed algorithm. Details are presented in the previous sections.

As a result of Proposition 2, each time we select a new seed, only the influence increments of nodes that have overlapping influence regions with the newly selected seed need to be re-evaluated. Formally, we define the set of Peer Seeds (PS) of a vertex $v \in V$ as follow:

$$PS(\mathcal{G}, v, \theta) = \{s \in V | MIOA(\mathcal{G}, s, \theta) \cap MIOA(\mathcal{G}, v, \theta) \neq \emptyset\}. \tag{4.11}$$

$PS(\mathcal{G}, v, \theta)$ can be computed efficiently just once at the beginning when all $MIOA(\mathcal{G}, v, \theta)$'s are available.

Combining the ideas of 1) limiting the region to be re-evaluated using PS, 2) limiting the set of nodes to pick from (adopted from CELF), and 3) picking nodes w.r.t its cost and the remaining budget (Algorithm 4), we have the complete procedure to determine the optimal seed set in Algorithm 8. Figure 4.15 gives the block diagram of the proposed algorithm.

The seed selection algorithm proceed as follow: In the initialization phase (lines

Algorithm 8: The Proposed Algorithm

```
   input : network graph G(V, E) and budget b
 1 S = S_1 = s_max = ∅, σ_0 = 0, θ = influence threshold
 2 foreach v ∈ V do
 3     build MIOA(G, v, θ)
 4     D(v) = MIOA(G, v, θ)
 5     calculate σ(v) (LBP or Algorithm 5)
 6     δ(v) = σ(v)/c(v)
 7     δ_old(v) = 0
 8 build PS(G, v, θ), ∀v ∈ V
 9 s_max = arg max_{v∈V} σ(v)
10 while true do
11     u = arg max_{v∈V\S}(δ(v))
12     if c(S_1 ∪ u) ≤ B then
13         S_1 = S_1 ∪ {u}
14         σ_0 = σ(S)
15         δ_old(v) = δ(v), ∀v ∈ V\S_1
16         b = b − c(u)
17         δ_max = 0
18         foreach v ∈ PS(G, u, θ)\S_1 do
19             if δ_old(v) > δ_max then
20                 build D(S_1 ∪ {v}) (Algorithm 6 and 7)
21                 calculate σ(S_1 ∪ {v}) (LBP or Algorithm 5)
22                 δ(v) = (σ(S_1 ∪ {v}) − σ_0)/c(v)
23                 if δ(v) > δ_max then
24                     δ_max = δ(v)
25     V = V\u
26     if V = ∅ or b = 0 then
           break
27 S = arg max (σ(S_1), σ(s_max))
   output: selected seed set S
```

1 – 8), $MIOA$'s and PS'es are constructed. The second candidate solution s_{max} can be determined in $O(n)$ time (line 9). S_1 is computed by executing the loop in lines 10 – 26. Each node in V is ranked by its incremental spread-cost ratio and can be added to S_1 just once. The node with the highest ratio is included in S_1 if it does not violate the budget b (line 12), and the corresponding nodes will be re-evaluated (lines 18 – 24). The procedure terminates once all nodes were considered, or no more budget remains (line 26). Finally, the algorithm compares the spread of S_1, s_{max} and returns the solution with the larger spread.

Computation complexity: Recall that we denoted by n_0 the largest number of vertices, and by d the largest in-degree of a node in a DAG. For each node $v \in V$ in the initialization phase, building $MIOA(\mathcal{G}, v, \theta)$ takes $O(n_0 \log n_0)$, and estimating $\sigma(v)$ takes $O(n_0 d)$ using SPBP and $O(n_0 2^d)$ using LBP, respectively. Thus, depending on the algorithm used, the running time of initialization is $O(nn_0(\log n_0 + d))$ or $O(nn_0(\log n_0 + 2^d))$.

Let k be the number of seeds selected in the main loop (lines 10 – 26) and v_0 be the cardinality of the largest set of peer seeds, namely, $v_0 = \max_{\forall v \in V}\{|PS(\mathcal{G}, v, \theta)|\} = O(n_0)$. Therefore, nodal influence spread is updated $O(kn_0)$ times. Note that this is much less than the number of updates required by Algorithm 1 ($O(n^2)$) as we do not naively re-evaluate every node. Each time when the influence spread is updated, we need to rebuild the DAG (line 20 – takes $O(n_0 \log n_0)$ with DAG 1 or $O(n_0)$ with DAG 2) and calculate the influence spread (line 21 – takes $O(n_0 2^d)$ with LBP or $O(n_0 d)$ with SPBP). The total computation complexity for different combinations of algorithms is summarized as follows:

	DAG 1	DAG 2
LBP	$n_0(n + kn_0)(\log n_0 + 2^d)$	$n_0(2^d(kn_0 + n) + n \log n_0)$
SPBP	$n_0(n + kn_0)(\log n_0 + d)$	$n_0(n(\log n_0 + d) + kn_0(1 + d))$

Clearly, combining DAG 1 and LBP incurs the highest complexity while the combination of DAG 2 and SPBP is the fastest. From the analysis, it is easy to see that the computation complexity depends on n_0 and d. The proposed approach is more efficient with smaller n_0 and d; that is, when the graph is sparse and the edge

propagation probabilities are small, both are likely true in social networks.

4.4.7 Evaluation

In this section, we evaluate the performance of the proposed framework. First, implementation details and experimental setup are introduced. Then, we present the results on 1) performance on real-world social networks and 2) impact of network structures using synthetic graphs.

4.4.7.1 Experiment setup

The algorithms and implementation: In addition to the two DAG models and two methods to compute influence spread (a total of 4 combinations DAG1–LBP, DAG1–SPBP, DAG2–LBP, and DAG2–SPBP), we make comparison with the following algorithms:

- *PMIA(θ)* [28]: a fast heuristic algorithm that builds a tree-like structure model on which influence is spread. θ is the influence threshold. We will set $\theta = 1/160$ in all experiments as it was reported to yield the best performance. The PMIA implementation provided by the authors is optimized for IM, and thus its performance for BIM is excluded.

- *Greedy/CELF*: The greedy approach from [70] with CELF optimization in [78]. The number of simulation rounds for each $\sigma(\cdot)$ estimation is 10,000.

- *Weighted Degree*: The simple heuristic that selects k seeds that have maximum total out-connection weight. Weighted Degree has been reported to be working very well in practice.

We do not compare with other heuristics such as SP1M, SPM [73], PageRank [17], Random, DegreeDiscountIC [29] or Betweenness centrality [48] since they have been reported in previous studies [70, 28] to be either unscalable or have poorer performance.

We have implemented the proposed algorithms in C++. All experiments are conducted on a workstation running Ubuntu 11.04 with an Intel Core i5 CPU and 2GB memory. In order to implement LBP algorithm, we use libDAI [89] and Boost [1] libraries. We find out through the implementation that running LBP on networks with high in-degree nodes is very costly. Therefore when running LBP, we prune incoming edges on high in-degree nodes such that only ten edges with the highest propagation probabilities are retained. The implementation of PMIA is obtained from its authors. Note that with code optimization, the running time of our algorithms can be further reduced.

Datasets: We use four real-world network datasets from [9] and [115] to compare performance of different algorithms. The four datasets were selected so as they are representative of the structural features of large-scale social networks, and are of different scales – from several thousands to millions of edges. The first one is an email exchange network in a research lab, denoted by *Email*. Each researcher is a vertex and an email from a researcher u to v constitutes an edge. The second

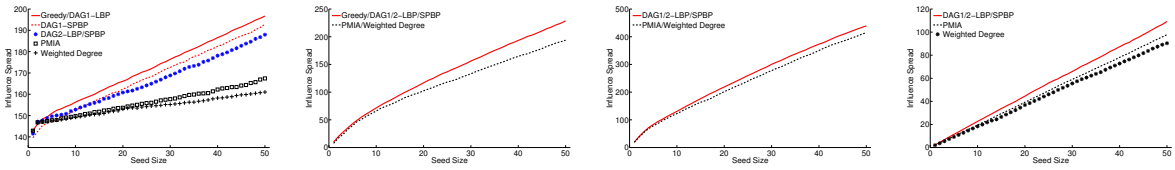
Table 4.4: Network datasets

Name	Nodes	Edges	Description
Email	447	5,731	Email communication within a research lab during a year
p2p-Gnutella	6,301	20,777	Gnutella peer to peer network from August 8 2002
soc-Slashdot	82,168	948,464	Slashdot social network from February 2009
Amazon	262,111	1,234,877	Amazon product co-purchasing network from March 2 2003

network, denoted by *p2p-Gnutella* is a snapshot of the Gnutella peer-to-peer file sharing network from August 2002. Nodes represent hosts in the Gnutella network and edges represent connections between the Gnutella hosts. The third network comes from Slashdot.org, a technology-related news website, denoted by *soc-Slashdot*. In 2002, Slashdot introduced the Slashdot Zoo feature that allows users to tag each other as friends or foes. The network contains friend/foe links between Slashdot users obtained in February 2009. Finally, *Amazon* dataset is the product co-purchasing network collected by crawling Amazon website on March 2, 2003. Details of the datasets are summarized in Table 4.4.

In addition to real social networks, we modified DIGG [39] source code and generated scale-free networks with different network densities and node out-degree distributions. It allows us to study the impact of graph structures and network property on the algorithm performance.

Probability generation models: Two models that have been used in previous work [70, 28] are: 1) the Weighted Cascade (WC) model where $p(u, v) = 1/d(v)$



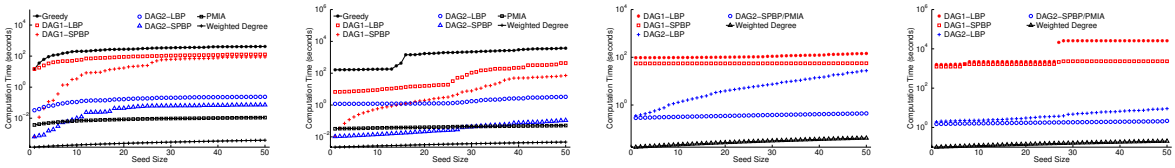
(a) *Email* (b) *p2p-Gnutella* (c) *soc-Slashdot* (d) *Amazon*

Figure 4.16: Influence spread with node unit-cost on 4 datasets. DAG 1 results are in red curves, DAG 2 are in blue curves, and other methods are in black curves.

where $d(v)$ is the in-degree of v and 2) the Trivalency (TV) model where $p(u, v)$ is assigned a small value for any $(u, v) \in E$. We argue that both models are not truthful reflections of the probability model in practice. The WC model assign a very high probability for a connections to nodes with small number of incoming connections while the TV model assigns a similar probability to all edges. In the evaluation, we consider two additional models: 1) Random (RA) where $p(u, v)$ is randomly selected in the range [0.001, 0.2]. RA is useful when no prior information regarding the influence is available; and 2) Power Law (PL) where $p(u, v)$ follows the power law distribution with the density function $p(x) = \alpha/x^{\beta}$, with x be the propagation probability between two random edges $p(u, v)$. Parameters $\alpha = 0.05$ and $\beta = 0.9$ are selected so that $p(u, v)$ has the mean value 0.1 in the range [0.001, 0.2].

4.4.7.2 Real social networks

Unit-cost version of BIM: BIM with unit-cost is the traditional IM problem where the seed set size k is fixed. In this experiment, we run 7 algorithms: Greedy, PMIA, Weighted Degree, and the 4 proposed methods on 4 datasets presented in



(a) *Email* (b) *p2p-Gnutella* (c) *soc-Slashdot* (d) *Amazon*

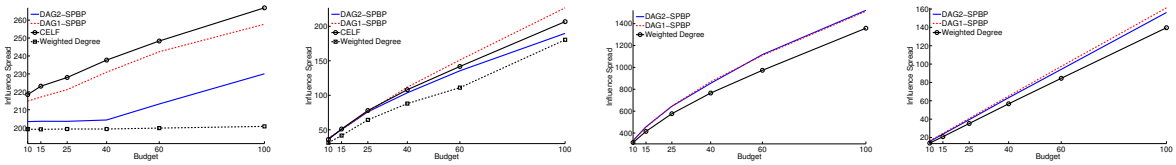
Figure 4.17: Computation time with node unit-cost on 4 datasets.

Table II. k varies from 1 to 50, and we adopt the RA probability generation model.

Figure 4.16 shows the influence spread generated by the best seed sets in different algorithms as the seed size changes. Since Greedy does not scale with large datasets, we only run Greedy on *Email* and *p2p-Gnutella*. The influence spread from the seed set selected by each algorithm is determined by 10,000 rounds of Monte Carlo simulations on the original graphs.

In Figure 4.16(a), the performance of DAG1–LBP and Greedy (known to be within a constant ratio of the optimal) are not distinguishable (and thus are represented in one curve). The influence spread of DAG1–SPBP and DAG2–LBP/SPBP are slightly behind, all outperforming PMIA and Weighted Degree. We observe on *Email* dataset (a small but dense network) that both the structure of the DAG (DAG 1 vs. DAG 2) as well as the BP algorithm used (LBP vs. SPBP) affect performance of the proposed methods. In contrast, as shown in Figure 4.16(b) – (d), the influence spreads of the four approaches DAG1/2–LBP/SPBP are identical for sparser networks, and is the same as Greedy in *p2p-Gnutella* dataset.

In terms of running time, Weighted Degree is the fastest. Among the four proposed approaches, DAG2–SPBP is the fastest, followed by DAG2–LBP, DAG1–SPBP, and finally DAG1–LBP. DAG2–SPBP and PMIA have comparable order in



(a) *Email* (b) *p2p-Gnutella* (c) *soc-Slashdot* (d) *Amazon*

Figure 4.18: Influence spread with random node costs on 4 datasets.

running time with DAG2–SPBP being 30-40% slower than PMIA in most cases. Again, this may be primarily attributed to the lack of code optimization in our proposed methods.

Interestingly, influence spread on *Amazon* grows linearly with the seed size. Our result matches with that in [28]. This can be explained by the sheer scale of the network, and thus the small number of selected seeds are likely to have non-overlapping influence regions.

General cost version of BIM: In this set of experiments, we compare only 4 algorithms: Greedy/CELF, Weighted Degree, and DAG1/DAG2–SPBP on 4 datasets presented in Table II. We also omit the two methods that use LBP (DAG1/DAG2–LBP) from the results since they have comparable performance as the SPBP approaches. The budget $b = \{10, 15, 25, 40, 60, 100\}$, and the RA probability generation model is chosen. Nodal costs are set uniformly in $[1.0, 3.0]$.

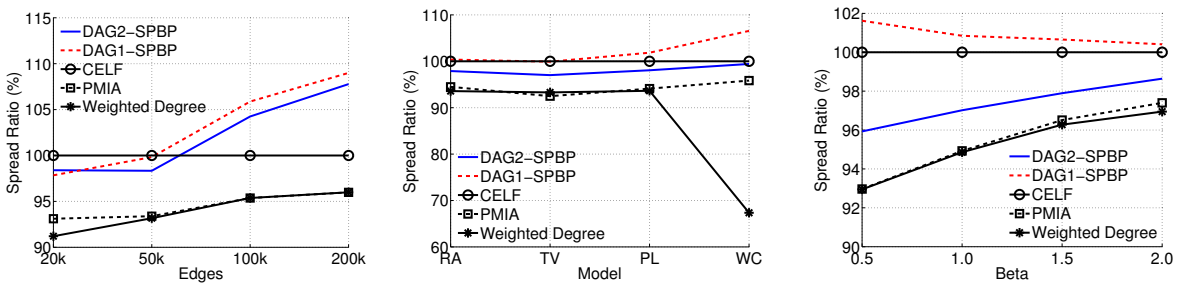
Results in Figure 4.18 are similar to that in Figure 4.16. In most cases, DAG1 has better performance compared to DAG2. Notably, DAG1–SPBP outperforms Greedy/CELF on *p2p-Gnutella* dataset. Results on running time (omitted due to space limit) show that the proposed methods are several orders of magnitude faster than Greedy/CELF. Weighted Degree while being the fastest algorithm, does not

perform nearly as well as the others on the dense graph (*Email*).

4.4.7.3 Synthetic networks

In this section, we conduct three sets of experiment with 5 methods: CELF, PMIA, Weighted Degree, and DAG1/2–SPBP. Synthetically generated networks are used to study the impact of network structures and probability generation models on performance of the algorithms. To isolate the effects of network properties, we only consider the unit cost BIM problem.

Impact of network density: Results from Figure 4.16 and 4.18 indicate that our proposed methods perform best on dense networks (*Email* and *p2p-Guntella*). To further validate this observation, we generate 4 networks with 20k, 50k, 100k, and 200k edges using DIGG [39]. The number of vertices is fixed at 5,000. We choose seed set size $k = 50$ and RA probability model. We evaluate the spread ratio of various algorithms, defined as the ratio of the spread attained to that by Greedy/CELF algorithm. From Figure 4.19(a), as the network density increases, the performance gap between the proposed algorithms and existing algorithms including CELF increases. CELF relies on many rounds of simulations to determine the spread. For dense networks, more rounds of simulations are needed to produce a spread estimation that is close enough to the *ground truth*. As a result, with a fixed number of simulation rounds, CELF has worse performance at high network densities. We also observe that PMIA, which was designed to take advantage of network sparsity; and Weighted Degree, which only uses local node information, do not perform well on densely connected graphs.



(a) Varying network densities (b) Varying prob. gen. models (c) Varying out-deg. distributions

Figure 4.19: Algorithm performance on different network conditions.

Impact of probability generation model: In this set of experiments, we run 5 algorithms on a synthetic network with 5,000 nodes and 50,000 edges. Each algorithm selects a seed set with size $k = 50$ under 4 propagation probability models: RA, TV, PL and WC. All models give similar performance except Weighted Degree on WC model. Recall that WC generates the propagation probabilities based on the in-degree of nodes, thus strong connections are established between nodes with low in-degree. Weighted Degree cannot "see" those strong ties beyond the local edges, and therefore, has the worst performance.

Impact of node out-degree distribution: It is known that node out-degree in real social networks follows the power-law distribution [34]. Let y be the percentage of nodes with degree x, then we have $y \sim \alpha/x^{\beta}$. α and β can be seen as the intercept and the (negative) slope when the sequence of nodes' degree is plotted on a log-log scale. While varying α only scales the distribution up or down, changing β alters the "shape" of the distribution. More specifically, a high value of β means the node out-degree distribution exhibits larger skew. The network in this case contains few "hubs" that are connected to many other nodes. On the other hand, a small β

means that the distribution is fat-tailed and the max out-degree in the network is not much larger than the average out-degree. We run 5 algorithms to solve the unit-cost BIM problem on 4 generated networks with $\beta = \{0.5, 1.0, 1.5, 2.0\}$. The network size is 5,000 and α is adjusted accordingly such that the total number of edges is roughly 50,000. We see from Figure 4.19(c) that the performance gap among the algorithms reduces with larger β. This is because with a large degree distribution skewness, nodes with high out-degree (hub) will almost certainly be one of the best seed candidates (unless their costs are too high, which is not this case). Simple algorithms such as Weighted Degree can easily identify such hub nodes. On the other hand, when the network is more "flattened", more sophisticated algorithms are necessary.

4.4.7.4 Remarks

From the experiments results, we observe that Weighted Degree gives the best efficiency in terms of spread/complexity. However, its performance degrades significantly on dense networks or more heavy-tail distributed graphs. The same conclusion can be drawn for PMIA. Though faster than our algorithms, PMIA shows little improvement in term of attainable spread compared to Weighted Degree, except under the WC model. Our proposed schemes surpass the others in all the experimented datasets. They also offer more flexibility: one would apply the best performed algorithm (DAG1–LBP) on static networks (e.g.: network of connections between co-workers) to identify the most influential nodes, or apply the fastest algorithm (DAG2–SPBP) on rapidly changing communities (e.g.: network of connections

between people in a social group) to obtain immediate result.

4.5 Summary

In this chapter, we first studied inference problems from interaction traces on social networks. Many key observations regarding the network properties from user influence perspective were made, including strong reciprocity, homophily and hierarchy in following relationships. Most importantly, we realized that the information adoption of a twitterer depends heavily on the influenceability of his first neighbor who spread that information. The FI diffusion model was therefore proposed and verified on many Twitter communities. Those findings are important in many application domains like viral marketing and building recommender systems since they change some of the basic assumptions regarding influence propagation in online social networks that were adopted in the past.

In the second part, we focused on the BIM problem, which is a generalization of the traditional IM problem. The study on real world datasets and synthetic datasets with controllable network parameters demonstrated that the proposed algorithms have superior performance. Furthermore, we gained some insights on the choice of algorithms in trading computation complexity with performance given the network structure.

Chapter 5

Conclusion

5.1 Summary of contributions

We presented in this thesis our work towards obtaining a better understanding of the complex network hidden property and behavior. Our work is a combination of methodologies research through designing new analytical techniques for complex network, and applications of these techniques in real world communication and online social networks. A basic premise behind our study is that mining interactions on the network leads to understanding its underlying structure and dynamics. We collected a large amount of network interaction data, performed extensive analysis and devised inference solutions. The key contributions that we made can be divided into two parts.

Part 1. On communication networks: We studied the inference problems from interaction traces on communication networks. We found a family of inference problems on networking domain that can be abstracted as functional relation in a bipartite graph where two separated set of nodes communicate with each other. We devised the binary independent component analysis with OR mixtures algorithm (bICA) to solve those problems. Given a set of data from the set of observable nodes and the assumption that hidden source nodes are statistical independent, bICA infers the connections between the two set of nodes and activities at each time slot of the source nodes. Furthermore, we proposed several optimization techniques to reduce the complexity of bICA. Finally, we demonstrated the usage of bICA through a wide range of networking applications, including monitoring in wireless networks, primary user separation in cognitive radio networks, multicast tree topology inference, and wireless transmission technology identification.

- **Optimal monitoring in multichannel wireless networks:** By assuming user activities are independent, we could apply bICA to find the best way to assign sniffers to channels so as the amount of observed data is maximized.
- **Primary user separation in cognitive radio networks:** We showed that the PU separation problem can be solved by gathering observations from all the SUs and applying bICA on top of the observation dataset.
- **Multicast tree topology inference:** We devised a variation of bICA, named seqBICA, which is specialized in inferring the hidden multicast topology from network end-points.

- **Binary blind identification of wireless transmission technologies:** We formulated a new and interesting problem of identifying transmissions technologies just by observing the spectrum sensing data. Experiment results showed that bICA can identify those hidden wireless transmissions with high accuracy, albeit the observation data are noisy.

Part 2. On social networks: The main contribution of this part is two-fold. We first conducted an extensive analysis on Twitter and influence measurement services and made several observations on the network characteristics from user influence perspective, including strong reciprocity, homophily and hierarchy in following relationships, and first-influencer diffusion model.

- **Strong reciprocity between users in a community:** showing that users that share similar interests do not randomly follow each other, but know each other well enough to form strongly connected network components.
- **Hierarchy in following relationship:** Following relationship encodes the hierarchial information on Twitter, where less influential users tend to follow those with higher influence.
- **Homophily in following relationship:** Homophily exists on Twitter from the user influence perspective, indicated by the fact that users with equal influence tend to be mutual friends.

- **First-influencer diffusion model:** We first observed that the success of information propagation on Twitter depends on the influence of the first information source. Then we proposed the FI model to formulate the spreading process on Twitter.

Additionally, we investigated the influence maximization (IM) problem. We introduced the budgeted influence maximization (BIM) problem, which is a generalization of the IM problem. We first showed that the problem is NP-Hard and a naively greedy approach does not work like on the IM problem. Then we proposed our improved greedy algorithm and proved that it provides a $(1 - 1/\sqrt{e})$-approximation for the BIM problem. Furthermore, we presented several techniques that can improve the seed selection and expedite the spread computation process. Combining everything together, we have a general framework in which many different belief-propagation algorithms and DAG models can be plugged-in to solve the BIM problem. Experiment results on both real-world social networks and synthetically generated networks showed the superior performance of the proposed solutions over state-of-the-art algorithms.

5.2 Future work

Next, we discuss several future research directions. Most of these problems arise during the course of research, but are yet to be investigated due to time limitation.

Practical implementation of bICA in wireless monitoring systems: In Section 3.6.1, we discussed the optimal solution for sniffer-channel assignment given fixed sniffer locations. Implementation of such a system should incorporate the learning procedure proposed in [12]. The time granularity of channel assignment should be sufficiently long to amortize the cost due to channel switching. To allow a consistent view of the channel at different locations, clock synchronization across multiple sniffers is needed. While clock synchronization can be performed offline using the frame traces collected [32], the accuracy of clock synchronization directly affects the inference accuracy of the ICA based methods in the sniffer-centric model. The choice of the slot of the binary measurements shall be made that takes into account the persistence of user transmission activities.

The channel assignment in its current form is computed in a centralized manner. This is reasonable since the sniffers are likely operated by a single administrative domain. An alternative distributed implementation has been considered in [13] for the user-centric model based on the annealed Gibbs sampler. However, parameters of the distributed algorithm need to be properly tuned for fast convergence (and hence less message exchanges). The sniffer-centric model is not immediately amiable to distributed implementation. It is therefore interesting to see if we can combine all existing components into a complete functional system.

bICA algorithm improvement: Even though bICA has been shown to perform very well on various networking problem, its complexity is still high with regard of the network size. Direct application of bICA in its current form to solve inference problems on large-scale networks is not feasible. We present in [91] the technique of

grouping similar rows in the mixing matrix $\boldsymbol{G}$, which was shown to greatly improve the efficiency of bICA. However, this technique can not be applied to other applications since rows on the mixing matrix are different in general. Another direction is modify bICA so as to easily incorporate *a priori* knowledge of the structure or active probabilities of latent variables. Examples of *a priori* knowledge of structural properties include small-diameter, scale-free, and community structure on complex networks. Furthermore, it is of intent to apply bICA on dynamic networks, where the set of nodes and edges "slightly" change after each time slot.

Application of the FI model on Twitter: In this dissertation, we made several discoveries on Twitter, including a new information diffusion model (FI). For the next step, we plan to apply the FI model to infer the edge propagation probabilities and solve the BIM problem on Twitter. We also believe that the following factors strongly affect the spread probability between two Twitter nodes, which warrants more investigation:

- **Mutual topical interest:** If two nodes share many topical interest, a message from one node will have more chance to be retweeted by the other. Topical interest of a user can be extracted from the content of its tweets. Such technique has been proposed in literature [114].

- **Tweet content quality:** A tweet with high quality and interesting contents generally has more chance to spread on the network. However, assessing the content quality is an open problem by itself. A naive solution would be to count the amount of favorites a tweet can get (similar to "like" on Facebook).

- **Past activities:** Two users with close offline relationship tend to stimulate online interactions and information spread. Although the offline relationship is hard to obtain, it can be inferred from online interactions [108, 53].

We yet to verify the correctness of those findings and how much impact they have on edge propagation probability. We first need to collect more data to conduct the study with high confidence. We believe that the new study will reveal more interesting findings that are very useful in the edge probability inference problem and many other application domains.

Appendix A

List of Publications and Manuscripts

A portion of the research presented in this thesis has appeared (or will appear) in refereed conferences and journals or currently in preparation. Here, we list the relevant publications and manuscripts associated with each topics

Inference on communication networks

- Huy Nguyen and R. Zheng, "A Binary Independent Component Analysis Approach to Tree Topology Inference", *IEEE Transactions on Signal Processing* (**TSP**), accepted for publication.

- Huy Nguyen, G. Scalosub, and R. Zheng, "On Quality of Monitoring for Multi-channel Wireless Infrastructure Networks", *IEEE Transactions on Mobile Computing* (**TMC**), accepted for publication.

- Huy Nguyen, G. Zheng, Z. Han, and R. Zheng, "Binary Inference for Primary User Separation in Cognitive Radio Networks", In *IEEE Transactions on Wireless Communications* (**TWC**), accepted for publication.

- Huy Nguyen, and R. Zheng, "Binary Independent Component Analysis with OR Mixtures", In *IEEE Transactions on Signal Processing* (**TSP**), Vol 59, Issue 7, July 2011.

- Huy Nguyen, N. Nguyen, Z. Han and R. Zheng, "Binary Blind Identification of Wireless Transmission Technologies for Wide-band Spectrum Monitoring", In *Proc. of the 54th IEEE Global Communication Conf.* (**GLOBECOM**), December 5-9, Houston, Texas, USA, 2011.

- A. Chhetri, Huy Nguyen, G. Scalosub, and R. Zheng, "On Quality of Monitoring for Multi-channel Wireless Infrastructure Networks", In *Proc. of the 11th ACM Int. Symp. on Mobile Ad Hoc Networking and Computing* (**MOBIHOC**), September 20-24, Chicago, Illinois, USA, 2010.

- Huy Nguyen, R. Zheng, and Z. Han, "Binary is Good: A Binary Inference Framework for Primary User Separation in Cognitive Radio Networks", In *Proc. of the 5th Int. Conf. on Cog. Radio Oriented Wireless Networks and Comm.* (**CrownCom**), June 9-11, Cannes, France, 2010.

Inference on social networks

- Huy Nguyen and R. Zheng, "On Budgeted Influence Maximization in Social Networks", *IEEE Journal on Selected Areas in Communications - Special Series on Network Science* (**JSAC NS**), accepted for publication.

- Huy Nguyen, and R. Zheng, "Influence Spread in Large-Scale Social Networks – A Belief Propagation Approach", In *Proc. of the 23rd European Conf. on Machine Learning and Principles and Practice of Knowledge Discovery in Databases* (**ECML-PKDD**), September 24-28, Bristol, UK, 2012.

- Huy Nguyen and R. Zheng, "Social Influence Score: Towards a Standardized Influence Measurement", *In preparation.*

Miscellaneous

- Y. Huang, M. Esmalifalak, Huy Nguyen, R. Zheng, Z. Han, H. Li, and L. Song, "Bad Data Injection in Smart Grid: Attack and Defense Mechanisms", In *IEEE Communications Magazine* (**ComMag**), Vol 51, Issue 1, January 2013.

- M. Esmalifalak, Huy Nguyen, R. Zheng, and Z. Han, "Stealth False Data Injection using Independent Component Analysis in Smart Grid", In *Proc. of the 2nd IEEE Int. Conf. on Smart Grid Communications* (**SmartGridComm**), October 17-20, Brussels, Belgium, 2011.

- M. Esmalifalak, Huy Nguyen, R. Zheng, L. Xie, L. Song, and Z. Han, "Stealthy Attack Against Electricity Market Using Independent Component Analysis", *Submitted to IEEE Transactions on Smart Grid.*

Bibliography

[1] Boost c++ libraries, http://http://www.boost.org/.

[2] Clean tweets, https://addons.mozilla.org/en-us/firefox/addon/clean-tweets/.

[3] Community resource for archiving wireless data at dartmouth, http://crawdad.cs.dartmouth.edu/.

[4] Klout, http://klout.com/home.

[5] The official klout blog, http://corp.klout.com/blog/.

[6] Peerindex, http://www.peerindex.com.

[7] Root-mean-square deviation, http://en.wikipedia.org/wiki/root-mean-square-deviation.

[8] Social computing data repository at asu, http://socialcomputing.asu.edu/pages/datasets.

[9] Stanford large network dataset collection, http://snap.stanford.edu/data/.

[10] Traceroute, http://www.caida.org/tools/.

[11] Twitter kills the api whitelist: What it means for developers and innovation, http://readwrite.com/2011/02/11/twitter_kills_the_api_whitelist_what_it_means_for.

[12] P. Arora, C. Szepesvri, and R. Zheng. Sequential learning for optimal monitoring of multi-channel wireless networks. In *INFOCOM*, pages 1152–1160. IEEE, 2011.

[13] P. Arora, N. Xia, and R. Zheng. A gibbs sampler approach for optimal distributed monitoring of multi-channel wireless networks. In *Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE*, pages 1–6, 2011.

[14] E. Bakshy, J. M. Hofman, W. A. Mason, and D. J. Watts. Everyone's an influencer: quantifying influence on twitter. In *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining*, WSDM '11, pages 65–74, New York, NY, USA, 2011. ACM.

[15] M. Barthélemy and N. L. A. Amaral. Small-world networks: Evidence for a crossover picture. *Physical Review Letters*, 82:3180–3183, 1999.

[16] R. Belohlavek and V. Vychodil. Discovery of optimal factors in binary data via a novel method of matrix decomposition. *J. Comput. Syst. Sci.*, 76:3–20, February 2010.

[17] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. *Comput. Netw. ISDN Syst.*, 30:107–117, April 1998.

[18] B. Businessweek. Facebook: The making of 1 billion users, http://www.businessweek.com/articles/2012-10-04/facebook-the-making-of-1-billion-users, 2012.

[19] R. Cáceres, N. Duffield, J. Horowitz, F. L. Presti, and D. Towsley. Loss-based inference of multicast network topology. In *Proceedings 1999 IEEE Conference on Decision and Control*, pages 3065–3070, 1999.

[20] R. Cáceres, N. G. Duffield, J. Horowitz, F. L. Presti, and D. Towsley. Statistical inference of internal network loss and topology. *SIGMETRICS Perform. Eval. Rev.*, 27(3):5–6, 1999.

[21] J. Cao, A. Chen, and P. Lee. Modeling time correlation in passive network loss tomography. In *Dependable Systems Networks (DSN), 2011 IEEE/IFIP 41st International Conference on*, pages 550 –561, june 2011.

[22] J. Cao, S. V. Wiel, B. Yu, and Z. Zhu. A scalable method for estimating network traffic matrices from link counts. Technical report, Bell Labs, 2000.

[23] K. M. Carley, J. Diesner, J. Reminga, and M. Tsvetovat. Toward an interoperable dynamic network analysis toolkit. *Decis. Support Syst.*, 43:1324–1347, August 2007.

[24] R. Castro, M. Coates, G. Liang, R. Nowak, and B. Yu. Network tomography: recent developments. *Statistical Science*, 19:499–517, 2004.

[25] M. Cha, H. Haddadi, F. Benevenuto, and K. P. Gummadi. Measuring user influence in twitter: The million follower fallacy. In *ICWSM 10: Proceedings of International AAAI Conference on Weblogs and Social*, pages 121–130, 2010.

[26] D. Chakrabarti, Y. Wang, C. Wang, J. Leskovec, and C. Faloutsos. Epidemic thresholds in real networks. *ACM Trans. Inf. Syst. Secur.*, 10(4):1:1–1:26, Jan. 2008.

[27] C. Chekuri and A. Kumar. Maximum coverage problem with group budget constraints and applications. In *Proceedings of the 7th APPROX*, pages 72–83, 2004.

[28] W. Chen, C. Wang, and Y. Wang. Scalable influence maximization for prevalent viral marketing in large-scale social networks. In *Proceedings of the KDD '10*, pages 1029–1038, New York, NY, USA, 2010. ACM.

[29] W. Chen, Y. Wang, and S. Yang. Efficient influence maximization in social networks. In *Proceedings of the KDD '09*, pages 199–208, New York, NY, USA, 2009. ACM.

[30] W. Chen, Y. Yuan, and L. Zhang. Scalable influence maximization in social networks under the linear threshold model. In *Proceedings of the 2010 IEEE ICDM '10*, pages 88–97, 2010.

[31] A. Chhetri, H. Nguyen, G. Scalosub, and R. Zheng. On quality of monitoring for multi-channel wireless infrastructure networks. In *Proceedings of the Eleventh ACM International Symposium on Mobile ad-hoc Networking and Computing*, MobiHoc '10, pages 111–120, New York, NY, USA, 2010. ACM.

[32] A. Chhetri and R. Zheng. WiserAnalyzer: A passive monitoring framework for wlans. In *Proceedings of the 5th International Conference on Mobile ad-hoc and Sensor Networks (MSN)*, 2009.

[33] A. Clauset, C. Moore, and M. E. J. Newman. Structural inference of hierarchies in networks. In *Proceedings of the 2006 Conference on Statistical Network Analysis*, ICML'06, pages 1–13, Berlin, Heidelberg, 2007. Springer-Verlag.

[34] A. Clauset, C. R. Shalizi, and M. E. J. Newman. Power-law distributions in empirical data. *SIAM Rev.*, 51(4):661–703, Nov. 2009.

[35] R. Cohen and S. Havlin. Scale-free networks are ultrasmall. *Phys. Rev. Lett.*, 90:058701:1–4, Feb 2003.

[36] R. Cohen, S. Havlin, and D. ben Avraham. *Handbook of Graphs and Networks*, chapter Structural Properties of Scale-free Networks, pages 85–110. Wiley-VCH Verlag GmbH & Co. KGaA, 2005.

[37] F. W. Computer and F. Wood. A non-parametric bayesian method for inferring hidden causes. In *Proceedings of the Twenty-Second Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 536–543. AUAI Press, 2006.

[38] K. Diamantaras and T. Papadimitriou. Blind deconvolution of multi-input single-output systems with binary sources. *Signal Processing, IEEE Transactions on*, 54(10):3720 –3731, October 2006.

[39] L. Dignan. Dynamic graph generator, http://digg.cs.tufts.edu/, 2006.

[40] P. Domingos and M. Richardson. Mining the network value of customers. In *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '01, pages 57–66, New York, NY, USA, 2001. ACM.

[41] Z.-B. Dong, G.-J. Song, K.-Q. Xie, and J.-Y. Wang. An experimental study of large-scale mobile social network. In *Proceedings of the 18th International Conference on World Wide Web*, WWW '09, pages 1175–1176, New York, NY, USA, 2009. ACM.

[42] N. Duffield. Network tomography of binary network performance characteristics. *Information Theory, IEEE Transactions on*, 52(12):5373 –5388, dec. 2006.

[43] N. Duffield, J. Horowitz, F. Lo Presti, and D. Towsley. Multicast topology inference from measured end-to-end loss. *Information Theory, IEEE Transactions on*, 48(1):26 –45, jan 2002.

[44] N. Duffield, J. Horowitz, F. Lo Presti, and D. Towsley. Multicast topology inference from measured end-to-end loss. *Information Theory, IEEE Transactions on*, 48(1):26–45, 2002.

[45] N. Duffield, F. Lo Presti, V. Paxson, and D. Towsley. Inferring link loss using striped unicast probes. In *INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 2, pages 915 –923 vol.2, 2001.

[46] N. Duffield, F. Lo Presti, V. Paxson, and D. Towsley. Network loss tomography using striped unicast probes. *Networking, IEEE/ACM Transactions on*, 14(4):697 –710, Aug. 2006.

[47] Federal Communications Commission. Spectrum policy task force report. *Report ET Docket no. 02-135*, Nov. 2002.

[48] L. Freeman. Centrality in social networks conceptual clarification. *Social Networks*, 1(3):215–239, 1979.

[49] B. J. Frey, R. Koetter, and N. Petrovic. *Neural Information Processing Systems*, volume 14, chapter Very loopy belief propagation for unwrapping phase images, pages 737–743. MIT Press, 2001.

[50] A. A. Frolov, D. Húsek, I. P. Muraviev, and P. Y. Polyakov. Boolean factor analysis by attractor neural network. *IEEE Transactions on Neural Networks*, 18(3):698–707, 2007.

[51] D. Ghita, H. Nguyen, M. Kurant, A. Argyraki, and P. Thiran. Netscope: Practical Network Loss Tomography. In *Proceedings of the 29th IEEE Conference on Computer Communications (INFOCOM)*, pages 1262–1270, 2010.

[52] M. Girvan and M. E. J. Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 99(12):7821–7826, 2002.

[53] J. Golbeck and J. Hendler. Inferring binary trust relationships in web-based social networks. *ACM Trans. Internet Technol.*, 6(4):497–529, Nov. 2006.

[54] J. Goldenberg, B. Libai, and E. Muller. Talk of the network: A complex systems look at the underlying process of word-of-mouth. *Marketing Letters*, 12:211–223, 2001.

[55] M. Gomez Rodriguez, J. Leskovec, and A. Krause. Inferring networks of diffusion and influence. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '10, pages 1019–1028, New York, NY, USA, 2010. ACM.

[56] A. Goyal, F. Bonchi, and L. V. Lakshmanan. Learning influence probabilities in social networks. In *Proceedings of the Third ACM International Conference on Web Search and Data Mining*, WSDM '10, pages 241–250, New York, NY, USA, 2010. ACM.

[57] M. Granovetter. Threshold Models of Collective Behavior. *American Journal of Sociology*, 83(6):1420–1443, 1978.

[58] T. Griffiths and Z. Ghahramani. Infinite latent feature models and the indian buffet process. *SciencesNew York*, 18(GCNU TR 2005-001):475–500, 2005.

[59] I. Griva, S. G. Nash, and A. Sofer. *Linear and Nonlinear Optimization, Second Edition*. Society for Industrial Mathematics, 2nd edition, December 2008.

[60] Y. Gu, G. Jiang, V. Singh, and Y. Zhang. Optimal probing for unicast network delay tomography. In *INFOCOM, 2010 Proceedings IEEE*, pages 1–9, March 2010.

[61] M. Gupte, P. Shankar, J. Li, S. Muthukrishnan, and L. Iftode. Finding hierarchy in directed online social networks. In *Proceedings of the 20th International Conference on World Wide Web*, WWW '11, pages 557–566, New York, NY, USA, 2011. ACM.

[62] S. Haykin. Cognitive radio: brain-empowered wireless communications. *IEEE JSAC*, 23:201–220, Feb. 2005.

[63] D. Húsek, P. Moravec, V. Snásel, A. A. Frolov, H. Rezanková, and P. Polyakov. Comparison of neural network boolean factor analysis method with some other dimension reduction methods on bars problem. In *International Conference on Pattern Recognition and Machine Intelligence*, pages 235–243, 2007.

[64] D. Húsek, H. Rezanková, V. Snásel, A. A. Frolov, and P. Polyakov. Neural network nonlinear factor analysis of high dimensional binary signals. In *The International Conference on Signal-Image Technology and Internet-based Systems*, pages 86–89, 2005.

[65] A. Hyvärinen and E. Oja. Independent component analysis: algorithms and applications. *Neural Netw.*, 13(4-5):411–430, 2000.

[66] A. Java, X. Song, T. Finin, and B. Tseng. Why we twitter: understanding microblogging usage and communities. In *Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 Workshop on Web Mining and Social Network Analysis*, WebKDD/SNA-KDD '07, pages 56–65, New York, NY, USA, 2007. ACM.

[67] D. Jin, B. Yang, C. Baquero, D. Liu, D. He, and J. Liu. A markov random walk under constraint for discovering overlapping communities in complex networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2011(05):P05031+22, 2011.

[68] JobStock. Social media statistics 2013 facebook vs twitter vs pinterest, http://www.jobstock.com/blog/social-media-statistics-2013/, 2012.

[69] A. Kabn and E. Bingham. Factorisation and denoising of 0-1 data: a variational approach. *Neurocomputing, special*, 71(10-12), 2009.

[70] D. Kempe, J. Kleinberg, and E. Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '03, pages 137–146, New York, NY, USA, 2003. ACM.

[71] D. Kempe, J. Kleinberg, and E. Tardos. Influential nodes in a diffusion model for social networks. In *Proceedings of the 32nd International Conference on Automata, Languages and Programming*, ICALP'05, pages 1127–1138, Berlin, Heidelberg, 2005. Springer-Verlag.

[72] S. Khuller, A. Moss, and J. S. Naor. The budgeted maximum coverage problem. *in Infor. Proc. Letters*, 70(1):39 – 45, 1999.

[73] M. Kimura and K. Saito. Tractable models for information diffusion in social networks. In *Knowledge Discovery in Databases: PKDD 2006*, volume 4213, pages 259–271. 2006.

[74] A. Krause and C. Guestrin. A note on the budgeted maximization of submodular functions. Technical Report CMU-CALD-05-103, Carnegie Mellon University - School of Computer Science, June 2005.

[75] B. Krishnamurthy, P. Gill, and M. Arlitt. A few chirps about twitter. In *Proceedings of the 1st Workshop on Online Social Networks*, WOSN '08, pages 19–24, New York, NY, USA, 2008. ACM.

[76] H. Kwak, C. Lee, H. Park, and S. Moon. What is twitter, a social network or a news media? In *Proceedings of the 19th International Conference on World Wide Web*, WWW '10, pages 591–600, New York, NY, USA, 2010. ACM.

[77] S. L. Lauritzen and D. J. Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society. Series B (Methodological)*, 50(2):157–224, 1988.

[78] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, and N. Glance. Cost-effective outbreak detection in networks. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '07, pages 420–429, New York, NY, USA, 2007. ACM.

[79] K. Lewis, J. Kaufman, M. Gonzalez, A. Wimmer, and N. Christakis. Tastes, ties, and time: A new social network dataset using facebook.com. *Social Networks*, 30(4):330–342, 2008.

[80] Y. Li, A. Cichocki, and L. Zhang. Blind separation and extraction of binary sources. *IEICE Trans. Fund. Electron. Commun. Comput. Sci.*, E86-A(3):580589, 2003.

[81] G. Liang and B. Yu. Maximum pseudo likelihood estimation in network tomography. *Signal Processing, IEEE Transactions on*, 51(8):2043–2053, 2003.

[82] A. S. Maiya and T. Y. Berger-Wolf. Inferring the maximum likelihood hierarchy in social networks. In *Proceedings of the 2009 International Conference on Computational Science and Engineering - Volume 04*, CSE '09, pages 245–250, Washington, DC, USA, 2009. IEEE Computer Society.

[83] Y. Mao, H. Jamjoom, S. Tao, and J. M. Smith. Networkmd: topology inference and failure diagnosis in the last mile. In *IMC '07: Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement*, pages 189–202, New York, NY, USA, 2007. ACM.

[84] R. McEliece, D. MacKay, and J.-F. Cheng. Turbo decoding as an instance of pearl's belief propagation algorithm. *JSAC*, 16(2):140 –152, Feb. 1998.

[85] M. McPherson, L. S. Lovin, and J. M. Cook. Birds of a feather: Homophily in social networks. *Annual Review of Sociology*, 27(1):415–444, 2001.

[86] B. Meeder, B. Karrer, A. Sayedi, R. Ravi, C. Borgs, and J. Chayes. We know who you followed last summer: inferring social link creation times in twitter. In *Proceedings of the 20th International Conference on World Wide Web*, WWW '11, pages 517–526, New York, NY, USA, 2011. ACM.

[87] S. Milgram. The small world problem. *Psychology Today*, 1(1):60–67, 1967.

[88] J. Mitola and G. Q. Maguire. Cognitive radio: Making software radios more personal. *IEEE Pers. Commun.*, 6:13–18, Aug. 1999.

[89] J. M. Mooij. libDAI: A free and open source C++ library for discrete approximate inference in graphical models. *Journal of Machine Learning Research*, 11:2169–2173, Aug. 2010.

[90] K. P. Murphy, Y. Weiss, and M. I. Jordan. Loopy belief propagation for approximate inference: An empirical study. In *Proceedings of Uncertainty in AI*, pages 467–475, 1999.

[91] H. Nguyen, N. Nguyen, G. Zheng, Z. Han, and R. Zheng. Binary blind identification of wireless transmission technologies for wide-band spectrum monitoring. In *Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE*, pages 1–6, 2011.

[92] H. Nguyen and P. Thiran. The boolean solution to the congested ip link location problem: Theory and practice. In *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, pages 2117 –2125, may 2007.

[93] H. Nguyen, G. Zheng, Z. Han, and R. Zheng. Binary inference for primary user separation in cognitive radio networks. *IEEE Transactions of Wireless Communications*. to be appeared.

[94] H. Nguyen and R. Zheng. A binary independent component analysis approach to tree topology inference. *IEEE Transactions of Signal Processing*. to be appeared.

[95] H. X. Nguyen and P. Thiran. Network loss inference with second order statistics of end-to-end flows. In *IMC '07: Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement*, pages 227–240, New York, NY, USA, 2007. ACM.

[96] N. Pathak, A. Banerjee, and J. Srivastava. A generalized linear threshold model for multiple cascades. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, pages 965–970, 2010.

[97] B. A. Prakash, H. Tong, N. Valler, M. Faloutsos, and C. Faloutsos. Virus propagation on time-varying networks: theory and immunization algorithms. In *Proceedings of the 2010 ECML PKDD: Part III*, pages 99–114, Berlin, Heidelberg, 2010. Springer-Verlag.

[98] M. Rabbat, M. Coates, and R. Nowak. Multiple-source internet tomography. *Selected Areas in Communications, IEEE Journal on*, 24(12):2221–2234, Dec. 2006.

[99] M. Rabbat, R. Nowak, and M. Coates. Multiple source, multiple destination network tomography. In *INFOCOM 2004. Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies*, volume 3, pages 1628–1639, March 2004.

[100] S. Ratnasamy and S. McCanne. Inference of multicast routing trees and bottleneck bandwidths using end-to-end measurements. In *INFOCOM '99. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 1, pages 353–360, 1999.

[101] M. Richardson and P. Domingos. Mining knowledge-sharing sites for viral marketing. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '02, pages 61–70, New York, NY, USA, 2002. ACM.

[102] R. Rowe, G. Creamer, S. Hershkop, and S. J. Stolfo. Automated social hierarchy detection through email network analysis. In *Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 Workshop on Web Mining and Social Network Analysis*, WebKDD/SNA-KDD '07, pages 109–117, New York, NY, USA, 2007. ACM.

[103] E. Sadikov, M. Medina, J. Leskovec, and H. Garcia-Molina. Correcting for missing data in information cascades. In *Proceedings of the 4th ACM International Conference on Web Search and Data Mining*, WSDM '11, pages 55–64, New York, NY, USA, 2011. ACM.

[104] K. Saito, R. Nakano, and M. Kimura. Prediction of information diffusion probabilities for independent cascade model. In I. Lovrek, R. Howlett, and L. Jain, editors, *Knowledge-based Intelligent Information and Engineering Systems*, volume 5179 of *Lecture Notes in Computer Science*, pages 67–75. Springer Berlin Heidelberg, 2008.

[105] G. R. M. A. A. L. Sales-Pardo, M. Extracting the hierarchical organization of complex systems. *Proc. Natl. Acad. Sci. U. S. A.*, 104:15224–15229, Sep. 2007.

[106] Y. Singer. How to win friends and influence people, truthfully: influence maximization mechanisms for social networks. In *Proceedings of the 5th ACM International Conference on Web Search and Data Mining*, WSDM '12, pages 733–742, New York, NY, USA, 2012. ACM.

[107] A. P. Streich, M. Frank, D. Basin, and J. M. Buhmann. Multi-assignment clustering for boolean data. In *ICML '09: Proceedings of the 26th Annual International Conference on Machine Learning*, pages 969–976, New York, NY, USA, 2009. ACM.

[108] W. Tang, H. Zhuang, and J. Tang. Learning to infer social ties in large networks. In *Proceedings of the 2011 European Conference on Machine Learning and Knowledge Discovery in Databases - Volume Part III*, ECML PKDD'11, pages 381–397, Berlin, Heidelberg, 2011. Springer-Verlag.

[109] H. Tian and H. Shen. Multicast-based inference for topology and network-internal loss performance from end-to-end measurements. *Computer Communications*, 29(11):1936 – 1947, 2006.

[110] H. Tong, B. Prakash, C. Tsourakakis, T. Eliassi-Rad, C. Faloutsos, and D. Chau. On the vulnerability of large graphs. In *IEEE 10th ICDM*, pages 1091 –1096, Dec. 2010.

[111] Twitter. Twitter 2012 trends, https://2012.twitter.com/en/trends.html, 2012.

[112] L. G. Valiant. The complexity of enumeration and reliability problems. *SIAM Journal on Computing*, 8(3):410–421, 1979.

[113] T. Šingliar and M. Hauskrecht. Noisy-or component analysis and its application to link analysis. *J. Mach. Learn. Res.*, 7:2189–2213, December 2006.

[114] J. Weng, E.-P. Lim, J. Jiang, and Q. He. Twitterrank: finding topic-sensitive influential twitterers. In *Proceedings of the 3rd ACM International Conference on Web Search and Data Mining*, WSDM '10, pages 261–270, New York, NY, USA, 2010. ACM.

[115] I. Wiki. Social network generation, http://www.infovis-wiki.net/index.php/social_network_generation.

[116] J. Yang and J. Leskovec. Modeling information diffusion in implicit networks. In *Proceedings of the 2010 IEEE International Conference on Data Mining*, ICDM '10, pages 599–608, Washington, DC, USA, 2010. IEEE Computer Society.

[117] J. S. Yedidia, W. T. Freeman, and Y. Weiss. *Exploring artificial intelligence in the new millennium*, chapter Understanding belief propagation and its generalizations, pages 239–269. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2003.

[118] A. Yeredor. Ica in boolean xor mixtures. In *Proceedings of the 7th International Conference on Independent Component Analysis and Signal Separation*, ICA'07, pages 827–835, Berlin, Heidelberg, 2007. Springer-Verlag.

[119] B. Yu, J. Cao, D. Davis, and S. Vander Wiel. Time-varying network tomography: router link data. In *Information Theory, 2000. Proceedings. IEEE International Symposium on*, pages 79–89, 2000.

[120] D. Zhao and M. B. Rosson. How and why people twitter: the role that micro-blogging plays in informal communication at work. In *Proceedings of the ACM 2009 International Conference on Supporting Group Work*, GROUP '09, pages 243–252, New York, NY, USA, 2009. ACM.